OPTIMAL MAINTENANCE FOR STOCHASTICALLY DEGRADING SATELLITE CONSTELLATIONS

THESIS

Timothy J. Cook, First Lieutenant, USAF

AFIT/GOR/ENS/05-05

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT/GOR/ENS/05-05

OPTIMAL MAINTENANCE FOR STOCHASTICALLY DEGRADING SATELLITE CONSTELLATIONS

THESIS

Presented to the Faculty
Department of Operational Sciences
Graduate School of Engineering and Management
Air Force Institute of Technology
Air University
Air Education and Training Command
in Partial Fulfillment of the Requirements for the
Degree of Master of Science in Operations Research

Timothy J. Cook, B.S.
First Lieutenant, USAF

March 2005

AFIT/GOR/ENS/05-05

OPTIMAL MAINTENANCE FOR STOCHASTICALLY DEGRADING SATELLITE CONSTELLATIONS

Timothy J. Cook, B.S.

First Lieutenant, USAF

Approved:

Jeffrey P. Kharoufeh — 1 MAR 05

Dr. Jeffrey P. Kharoufeh — Date

Thesis Advisor

Gilbert L. Peterson — Feb. 25, 2005

Dr. Gilbert L. Peterson — Date

Committee Member

AFIT/GOR/ENS/05-05

Abstract

This thesis develops a methodology to determine an optimal policy for maintaining a satellite constellation that degrades over time. Previous work has developed a methodology to compute an optimal replacement policy for a satellite constellation in which satellites were viewed as binary entities, either operational or failed. This research extends the previous models by developing an optimal maintenance policy for satellite constellations in which each satellite may operate in a finite number of degraded states. The constellation is assumed to consist of a finite number of satellites, each with a finite number of functions with distinct failure mechanisms. Assuming each function lifetime is exponentially distributed, the stochastic degradation process is modelled as a discrete-time Markov chain. The degradation process is subsequently used to formulate an optimization problem as a finite planning horizon Markov decision process in which the total expected loss of utility is minimized. The maintenance actions considered include on-orbit repairs and satellite replacements, each of which have an associated level of risk. Numerical examples are presented to illustrate the model, and a parametric sensitivity analysis is performed using notional data to determine conditions for which the resulting policy consists of only replacements, only on-orbit repairs, or mixtures of replacements and on-orbit repairs.

Acknowledgements

The research presented herein could not have been accomplished without the help of a number of people. I am particularly thankful for the thorough guidance of Dr. Jeff Kharoufeh throughout the entire research process. He provided invaluable insight into the problem that greatly improved the quality of this work. I would like to express my gratitude to my reader, Dr. Bert Peterson, who helped me to clearly express the results of my effort. I would also like to thank Mr. Justin Comstock of Welkin Associates, Ltd. for supporting this research effort and providing valuable insight into the "real world" problem. Finally, I would like to thank the entire March 2005 Operations Research class for spending valuable time discussing the problem with me.

I also would like to thank my wife for the loving support that she provided throughout the entire 18-month program. I particularly thank her for helping me correct grammatical and spelling errors in my thesis document. Special thanks also goes to my daughter who constantly reminded me of the reason for working so hard.

Timothy J. Cook

Table of Contents

List of Figures

List of Tables

OPTIMAL MAINTENANCE FOR STOCHASTICALLY DEGRADING SATELLITE CONSTELLATIONS

1. Introduction

1.1 Background

The world stepped into the space age in 1957 when the Soviet Union successfully launched the first satellite, Sputnik 1, into orbit. Not to be outdone by its Cold War enemy, the United States entered the space arena the following year when Explorer 1 orbited the earth for the first time, and the race to gain command of the ultimate high ground began. Ultimately, the Cold War (and the space race) would come to an end. However, its impact would prove to have a significant influence on the use of space assets for both military and commercial uses. Services provided by satellites such as communications, earth observation, weather observation, navigation, and many others have become vital to both public and private sector applications. In a national defense context, satellites are vital tools in intelligence gathering to help safeguard nations against enemy threats. Today, multiple satellites operate in clusters or *constellations* to provide such services to end users on a continuous basis.

The costs and benefits of operating a satellite constellation are enormous. Each satellite in a constellation may cost up to hundreds of millions of dollars to develop, build, and launch into orbit. However, the return on a satellite constellation can also be extremely high as billions of users receive service from satellite constellations on a daily basis. From a commercial standpoint, this results in billions of dollars in revenue each year (e.g., AT&T Wireless earned $4.21 billion in revenue in the

second quarter of 2004 [40]). From a military standpoint, the returns from a satellite constellation result in high levels of mission effectiveness.

The enormous potential returns from a satellite constellation indicate that any loss of capability can potentially be very costly. Satellites operate in a harsh environment and degrade over time, rendering the constellation less than fully capable as it ages. This can lead to decreased levels of end-user satisfaction which may result in lost revenue or lower mission effectiveness. Therefore, it is vital to formulate a strategy to mitigate the losses incurred as a result of a decrease in the quality of service. In order to develop such a strategy, it is necessary to determine acceptable levels of service loss and the time at which corrective action should be taken to improve the constellation's capability level. This is known as a maintenance policy. A maintenance policy may include such maintenance actions as preventive maintenance which is performed before a component fails to extend the component's lifetime, or reactive maintenance which is performed after a component fails to return it to a useful state. However, due to the inaccessibility of satellites, preventive maintenance may not be a viable option.

With the exception of the Hubble Space Telescope, satellite maintenance has traditionally consisted only of replacement actions. On-orbit repairs have been considered to be cost intensive and too risky to be a viable option for constellation maintenance. The tragedy of the Space Shuttle *Columbia* in 2003 has reinforced this idea. Current technological developments may provide a feasible option for satellite repairs in the future. Companies such as Advanced Industrial Science and Technology (AIST) [1] in Japan are developing space maintenance robots that may be capable of performing on-orbit repairs to satellites at a fraction of the cost of a manned mission. Thus, a major question that arises is: Can a maintenance policy be derived that will result in a minimum total loss, taking into account satellite maintenance costs and losses incurred due to a less than fully capable satellite constellation?

Satellite replacements in the public sector are currently driven by funding. More specifically, if the funding for a new satellite is available, the satellite is purchased. This can be a very costly method of maintaining a satellite constellation when the procurement cost outweighs the marginal improvement in the constellation's effectiveness. Billions of tax dollars may be unnecessarily spent on the purchase of satellites that do not significantly improve mission effectiveness. A more economically sound method of maintaining a satellite constellation is to purchase a replacement satellite under conditions for which the mission effectiveness improvement outweighs the cost of the replacement. Determining such conditions for other general multi-unit systems is a well-studied topic in the field of operations research, namely that of optimal maintenance policies.

A plethora of research exists concerning optimal maintenance policies for degrading systems. The most commonly employed policies are for single-unit, infinite planning horizon problems which generally incorporate three maintenance actions (do nothing, minimal repair, replace) and are obtained via Markov renewal theory. Optimal maintenance policies for more complex multi-unit systems are often obtained by means of stochastic dynamic programming using a discounting factor to determine a stationary policy. This stationary policy prescribes an optimal action for every state in which the system may be observed at any time. One drawback to this method is that a stationary policy does not always exist. When the system is too large and too complex to be modelled analytically, researchers have often resorted to computer simulation models to determine approximate maintenance policies.

The United States Government is facing the largest one-year budget deficit in history with forecasts as high as $477 billion in 2005. Some economists believe that continuing this spending trend could have a catastrophic effect on the economy. One of the largest areas of government spending is national security. However, due to the Global War on Terrorism, the President and Congress are unwilling to sacrifice national security to reduce the deficit. Even so, it may be possible to reduce

funding in this area without compromising national security. In order to accomplish this, system program managers must manage their systems more efficiently, thereby reducing their funding requirements. One way to reduce their funding requirement is to implement optimal maintenance policies for the systems they manage. One type of system for which this would have a significant effect is a satellite constellation due to the high costs involved and importance to national security. While it will by no means eliminate the budget deficit, implementing an optimal maintenance policy for satellite constellations may result in a significant reduction in government spending while simultaneously achieving national security objectives.

This research effort proposes a methodology that may be used by satellite constellation program managers to improve the cost effectiveness of maintenance operations. The result of the methodology is a maintenance policy that produces the minimum expected total cost over the planning horizon.

1.2 Problem Definition and Methodology

We consider a satellite constellation in which each satellite operates at a finite number of capability levels. The capability levels are determined by the number of functions[1] each satellite possesses. Each satellite's capability degrades over time (in the absence of maintenance) according to a stochastic degradation process. The satellites' degradation processes are assumed to be stochastically independent, but the cost structure for specific states of the satellites are economically dependent. The capability level of the satellite constellation is observed at discrete points in time through perfect inspections. Each time the constellation is observed, one of three maintenance actions is chosen for each satellite (do nothing, on-orbit repair, or replace) based on the expected cost that will result from the action taken and the stochastic degradation over the next inter-inspection interval. The costs associated

[1] A satellite function is defined in Section 3.1 to be an operation that must be performed for a satellite to successfully carry out its mission.

with the constellation are; a fixed satellite replacement cost, a state dependent on-orbit repair cost, and a state dependent penalty cost incurred when the constellation is found to be operating in a partially capable state at any inspection time.

The primary objective of this research is to provide a methodological framework that will prescribe a provably optimal maintenance policy for the satellite constellation described above. Previous research in the area of optimal satellite constellation maintenance policies considered satellites as either "up" or "down" with the only maintenance option being a satellite replacement. This thesis extends the previous research by allowing satellites to operate in partially capable states and considering multiple maintenance actions. An optimal maintenance policy for the satellite constellation requires a formally defined stochastic degradation process. This research will consider a completely observable capability level that is a function of the operational and non-operational functions of each satellite. The stochastic degradation level of a satellite constellation is assumed to evolve as a discrete-time Markov chain.

Once the stochastic degradation process is defined, an optimization problem is formulated to determine the optimal maintenance policy. The objective of the optimization problem is to minimize the total expected loss of operating a satellite constellation while accounting for both maintenance and penalty costs incurred while the constellation is in partially capable states. Due to the complexity and recursive nature of the problem, stochastic dynamic programming is used to determine the optimal policy over a finite planning horizon by formulating the the problem as a Markov decision process. A finite horizon is assumed for two reasons. First, in practice satellite constellation maintenance planning occurs over a finite horizon. Second, in a finite planning horizon at least one optimal policy will always exist, whereas there is no guarantee that an optimal stationary policy exists for an infinite horizon problem.

1.3 Thesis Outline

Chapter 2 provides a review of the pertinent literature with an emphasis on optimal maintenance models. Chapter 3 provides the formulation of the stochastic degradation process for a satellite constellation followed by a Markov decision process formulation which is subsequently used to obtain an optimal maintenance policy. Chapter 4 provides numerical illustrations using notional data and also presents a parametric sensitivity analysis. Chapter 5 provides concluding remarks and summarizes the main contributions of this research, and recommends the most fruitful areas of future research.

2. Review of the Literature

This chapter presents a review of the literature related to optimal maintenance policies for stochastically degrading systems. No literature could be found pertaining specifically to degrading satellite constellations; however, much research does exist in closely related subject areas. An investigation of current strategies for maintaining satellite constellations is presented in Section 2.1 with emphasis on measures of performance and budgeting strategies. Section 2.2 is a review of optimal maintenance policies for single unit systems and multi-unit systems. Section 2.2 also reviews maintenance models formulated as Markov decision processes.

2.1 Satellite Constellation Maintenance Planning

Current satellite maintenance planners generally consider only the launch of a new satellite as a viable maintenance action, as on-orbit repairs are considered to be both costly and risky. The decisions to launch new satellites typically come as a result of a resource allocation analysis in which cost is considered to be a constraint imposed by the national defense budget. If the money is available, a satellite may be purchased, but the problem is deciding which type of satellite to procure. A number of planning tools aide in making this decision. Some of these tools provide constellation performance measures to determine which constellations are most in need of a new satellite. Other tools provide an analysis of alternatives, returning an optimal procurement decision that meets budgetary and performance constraints.

2.1.1 Performance Measures

Satellite constellation performance measures play a large role in determining satellite acquisition schedules. Two types of measures are considered when determining a replacement policy. The first is a measure of satellite reliability, the probability that the satellite survives until some future time. The second is a measure of mission

effectiveness which describes the extent to which the constellation meets the needs of a certain mission (e.g., the percentage of time an end-user can receive service).

Until the late 1980's, it was widely accepted that satellite subsystem reliability could be modelled with an exponential distribution with failure rates provided by MIL-HDBK-217. Recent research argues that empirical data does not support a constant failure rate for satellite subsystems, but rather a "bathtub curve" failure rate such as the one in Figure 2.1. In his 1995 paper, Hansen [16] provides a methodology to determine what he claims is a more accurate and useful satellite reliability prediction. His method consists of performing Monte Carlo simulation using a five parameter Weibull distribution to model the lifetimes of each satellite subsystem. This five parameter Weibull distribution exhibits a "bathtub curve" failure rate. The result of his simulation is the distribution of the time to failure for the n^{th} ordered subsystem failure. The distribution of the number of subsystem failures over a given interval is also computed.

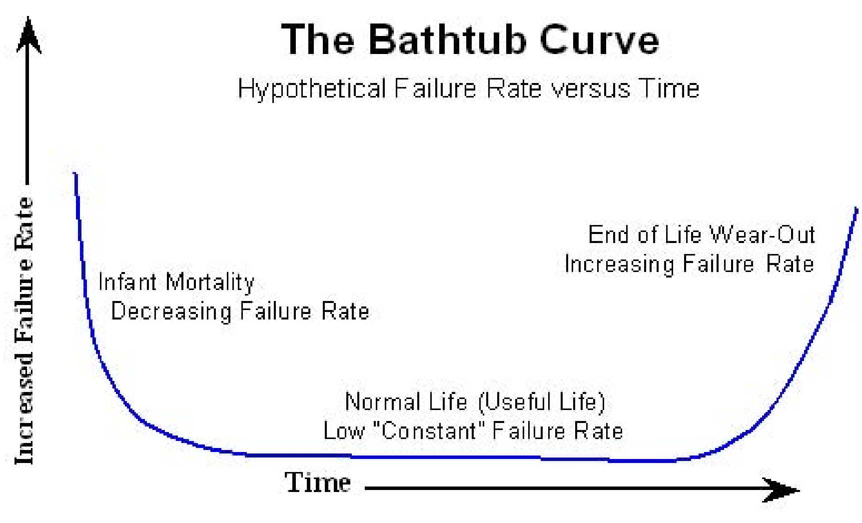

Figure 2.1 A hypothetical "bathtub curve" failure rate for a satellite. Adopted with permission from the ReliaSoft Corporation *http://www.weibull.com/hotwire/issue21/hottopics21.htm.*

Some performance measures may involve both a reliability measure and a mission effectiveness measure. One such model currently used is presented by Comstock [11]. This model measures the effectiveness of a satellite constellation based on a measure defined as the beginning of life (BOL) equivalence. Each satellite in the constellation is assigned a composite BOL equivalence score. The BOL equivalence score takes into account the satellite's reliability, duty cycle, and a subjective band performance. Each satellite BOL equivalence score is summed to determine the overall constellation BOL equivalence. The BOL is not relative to the performance level of the satellite being measured at its beginning of life, but is actually a measure related to the most technologically advanced satellite in the constellation. In other words, the BOL equivalence score is a ratio of the satellite's level of performance to the performance level of the newest satellite in the constellation. This allows the model to account for "degradation" due to technology improvements over time.

Another effectiveness measure is proposed by Wilson [45] who presents a model to measure the effectiveness of a satellite weather system that can be applied to other constellations as well. His model uses concepts from multivariate utility theory to develop a measure of effectiveness (MOE) to support decision making. This MOE is intended to measure the level of mission accomplishment for a constellation. The model divides the earth into shells that resemble the squares formed with latitudinal and longitudinal grid lines. Each of these shells is assigned a score ($MOE_{i,j}$) which is a multiplicative relationship between attributes based on their operating level and weight assuming utility independence among attributes. To determine the overall MOE he sums all the shell MOE's based on an assumption of additive utility:

$$MOE = \sum_i \sum_j MOE_{i,j}. \tag{2.1}$$

The weights used in the MOE computation are determined by a subject matter expert (SME) who *ranks* each of the attributes of a constellation that contribute

to a successful mission. Then the SME assigns each of the attributes a *rating* on $[0, 100]$. This permits the creation of a relative normalized weighting scheme for the attributes.

Another currently used method of determining the effectiveness of a satellite constellation is through combat simulations. An example of this type of tool is the System Effectiveness Analysis Simulation (SEAS) [37]. SEAS is an agent-based simulation used to quantify the contributions of services provided by all space systems including missiles, satellites, etc. Measures of this type often influence military satellite procurement decisions because it is a measure of the combat effectiveness of current space assets.

2.1.2 Satellite Acquisition Strategies

The current strategy of military space budget planners is to spend all of the money they are allocated in the most effective manner. The reasoning behind this is that under-utilized resources are likely to be retracted at the end of the fiscal year, and future funding potentially reduced. Decision makers often fear that if their funding is reduced, they will not be able to meet mission performance objectives in the future. Therefore, when considering the maintenance of a satellite constellation, military space budget planners attempt to answer the question, "How much can we do with the money we are allotted," rather than, "How can we continue to meet performance standards at a minimum cost."

One satellite funding allocation tool used by the military is the Space and Missile Optimization Analysis (SAMOA). Brown et al. [8] describe SAMOA as a five-component collection of tools used to determine which space systems will meet the requirements of Air Force Space Command over a 24-year period. The SAMOA toolkit was used to prepare the bi-annual Air Force Space Command's Strategic Master Plans in 1997 and 1999. The Space Command Optimizer of Utility Toolkit (SCOUT) is an important analysis tool within SAMOA. The authors

describe SCOUT as the mathematical analysis tool which determines the optimal mix of space systems to purchase (subject to numerous constraints) via integer linear programming. The constraints consist of budgetary and performance measures which may or may not be *elastic*, meaning they may be violated for an imposed penalty. The objective is to minimize the summation of these penalty costs, thereby meeting mission requirements and budgetary constraints as much as possible.

Another tool used to aide in determining satellite acquisition policies is the Operational Constellation Availability and Reliability Simulation (OSCARS). Jacobs et al. [17] provide a detailed description of OSCARS which was developed for use by the Air Force Space Command Launch Services Office. The purpose of OSCARS is to determine a satellite launch schedule that will maintain a specified number of operational satellites for a given constellation. When determining the launch schedule, OSCARS takes into consideration each satellite's lifetime distribution as well as the satellite production database and the booster production database. Any combination of satellite lifetime distributions may be assumed with OSCARS as the time-dependent distribution of the number of operational satellites is estimated using Monte-Carlo simulation.

2.2 Optimal Maintenance Policies

The techniques used to find analytical solutions to maintenance problems first appeared in the literature in the early 1900's in actuarial research concerning population control. By the 1930's, the industrial revolution had created a heavy reliance on expensive and unreliable industrial equipment. Those researchers studying population control problems began to recognize the utility of applying these actuarial concepts to industrial machinery. Lotka [21] was one of the first people do so when he used the population growth model framework to address the equipment replacement problem in 1939. Another early investigation into the replacement problem came in 1940 from Preinreich [31] who approached the problem from a purely eco-

nomic standpoint. Instead of considering the physical degradation of the machine, he considered the economic depreciation of the machine as the condition upon which to base replacements.

There are five basic components to any optimal maintenance policy model: the possible maintenance actions, the length of the planning horizon, the policy class, the optimization criteria, and the system characteristics. The possible maintenance actions may include such actions as minimal repairs which return the system to the state it was in just before failure or replacements which return the system to a new state. The planning horizon may be finite or infinite. An example policy class is an opportunistic maintenance policy in which maintenance is performed on multiple components of a system in one maintenance action. The optimization criteria is dependent on the planning horizon and may be to minimize the long-run average cost per unit time for an infinite horizon model, or it may be to minimize the total cost for a finite horizon model. The system characteristics include the type of failure process (degradation process or up down machine) and the type of state space (discrete or continuous, finite or infinite).

Optimal maintenance policies have been studied extensively; therefore, it is logical to begin with a review of some key surveys. McCall [24] provided the first extensive survey of maintenance models in his 1965 paper which classified maintenance scheduling policies by their common underlying structures. Another significant work in the area of maintenance policies in the same era is the classic text by Barlow and Proschan [4] in 1965, *The Mathematical Theory of Reliability*, in which the basic mathematical framework for various maintenance policies is rigorously reviewed. Pierskalla and Voelker [30] provided another extensive survey of maintenance models in 1975, emphasizing works subsequent to the McCall survey. The 1981 review by Sherif and Smith [36] provides a brief description of each type of maintenance policy and classifies 524 major works by the subject categories. In 1989, Valdez-Flores and Feldman [42] provided a survey of maintenance models with emphasis on the

single-unit models since the Pierskalla and Voelker survey. The 1991 survey by Cho and Parlar [10] complements the Valdez-Flores and Feldman survey by restricting attention to the multi-unit models since the Pierskalla and Voelker survey. The most recent survey on this subject is that of Wang [44] in 2002 in which he provides an updated classification of policies with an emphasis on single unit systems.

Maintenance policies appear in the literature for two general types of systems: maintenance policies for single unit systems, and maintenance policies for multi-unit systems. Optimal maintenance policies for single-unit systems are generally found using Markov analysis techniques such as Markov renewal theory to optimize some steady-state criterion such as the long-run average cost per unit time or the long-run availability of the system. Optimal maintenance policies for multi-unit systems are much more complex and may require the use of dynamic programming or simulation for large problems. Of the two types of policies, the multi-unit policies are most relevant to this research; however, a brief overview of single-unit policies will add to the understanding of multi-unit policies. Furthermore, a review of some maintenance policies found using Markov decision processes will also be provided.

2.2.1 Single-Unit Policies

The age replacement policy, the control-limit (sometimes referred to as wear-limit) policy, and the inspection policy are three of the most commonly studied maintenance policies for single-unit systems. These types of policies and their extensions are covered in great detail in the literature. Some of the more complex models use a hybrid policy which combines two or more of these policies for further precision.

One of the earliest and most frequently studied categories of single-unit policies is the *age replacement policy*. This type of policy calls for the instantaneous replacement of the system after operating for time T or at failure. Each time the system is replaced, the stochastic failure process renews because the system is returned to new

condition. Finding the optimal age replacement policy consists of determining an optimal operating time T at which a preventive replacement should occur such that some function is optimized. Barlow and Proschan [4:86] show that for the infinite time horizon problem, the optimal T is non-random. Barlow and Hunter [5] provide another optimal solution for this type of policy in which the limiting efficiency (fractional amount of time the system is up in the long run) is optimized. Nummelin [26] presents a more complex variant of the age replacement in which the replacement cost and the failure cost are dependent on the state of the system.

Morse [22] considers a variant of the age replacement problem in which perfect preventive maintenance or perfect repairs are performed on the system at time T or at failure, respectively. This has the same effect as replacing the system since the repairs return the system to the new state. The difference is that the repair times are non-negligible; the maintenance is not assumed to be instantaneous. In that model, the objective is to maximize the long-run net income from the system per unit time by finding the optimal T. Morse points out that preventive maintenance is only worthwhile if the system has an increasing failure rate and if the cost to perform maintenance at failure exceeds the cost to perform preventive maintenance.

The *control-limit* or *wear-limit policy* is similar to the age replacement policy except that time is replaced by some level of physical degradation (wear) which can be measured. Under this policy maintenance is performed when the wear level exceeds some threshold r, or on failure. When the system, is replaced, the stochastic wear process renews as in the age replacement policy models. Finding the optimal control-limit policy consists of determining the threshold r such that the desired performance measure is made optimal. Park [28] shows that a wear-limit replacement model in which the item's failure rate is dependent on some level of degradation closely resembles the age replacement model when the long-run total expected cost rate is minimized. A much more complex wear-limit policy model is proposed by Su et al.

[38] in which they consider multiple maintenance actions, each having an associated level of risk.

Another commonly studied category of single-unit maintenance policies is the *inspection policy*. Under an inspection policy, failures are only detected when the system is inspected, and maintenance is performed when the system is found in a failed state. A penalty is incurred for the time that a failure is left undetected. The objective of an inspection policy is to determine an inspection schedule such that some function of the penalties, maintenance costs, and inspection costs is minimized. Barlow et al. [6] present the basic inspection policy model from which numerous later works evolved. In this paper, only a single failure is considered and the penalty cost is dependent on the time that the failure goes undetected. An optimal inter-inspection time is determined such that the expected inspection plus failure costs are minimized. A more recent work concerning inspection policies is the work of Grall et al. [14]. In this paper, the authors present results for a complex inspection policy in which they consider a wear-limit policy simultaneously with an inspection policy to find the minimum long run expected maintenance cost per unit time. Bloch-Mercier [7] considers a similar strategy to [14] and shows that under some assumptions, the optimal long-run availability is produced by non-random inter-inspection intervals.

2.2.2 Multi-Unit Policies

Replacement policies for multi-unit systems need only be studied for systems in which there is a dependence between component lifetime distributions or if the system demonstrates economic dependence between components. In other words, if the components of a system are stochastically and economically independent, then a single-unit policy can be applied to each component to achieve an optimal system maintenance policy. Most of the literature in the area of multi-unit policies considers the case of economic dependence. With regard to the research presented in this

thesis, the most relevant classes of multi-unit maintenance models in the literature are the group maintenance policy, and the opportunistic maintenance policy.

The *group maintenance policy* is one of the most frequently studied multi-unit policies. In this type of policy, all components of a multi-unit system are replaced after reaching some set threshold such as time, number of component failures, or cumulative degradation. If a component fails prior to reaching the established threshold, minimal repair or replacement may be performed on that component. The group maintenance policy, in some aspects, is similar to the control limit or age replacement policies for a single-unit.

One of the earliest research efforts with regard to a group maintenance policy was the work of Campbell [9] in 1941. The author compares two maintenance policies for a series of street lamps. The first policy consists of replacing each lamp individually as it fails for an indefinite period of time. The second policy is a group replacement policy in which all lamps are replaced at times $\{nT : n = 1, 2, \ldots\}$ where T is the defined replacement interval. Additionally, if a lamp fails during this interval, it is individually replaced. Campbell assumes that the lamps each exhibit an increasing failure rate and is able to find values of T for which it is economically beneficial to choose the group maintenance policy.

Gertsbakh [13] proposes a group maintenance policy for a system consisting of n-independent, identical components having exponentially distributed lifetimes. In this article, group maintenance consists of replacing all failed components and is performed when k failures first occur. Each time the group maintenance is performed, a fixed setup cost plus the replacement cost of each individual component is incurred. Only the failed components are replaced because the lifetimes are assumed to be exponentially distributed meaning that replacing a component that has not failed will not reduce the chance that the component fails in the next instant. Gertsbakh shows that, under the optimality criterion of minimum cost per unit time and maximum return per unit time, the previously described group maintenance policy is optimal.

A recent work in group maintenance policies is that of Gürler and Kaya [15]. They consider a system in which each component may be classified as good, doubtful, preventive maintenance due (PM due), and down. Furthermore, each classification has a finite number of degradation levels within it. They consider a group maintenance policy in which the system is replaced when the number of doubtful components reaches a set threshold and another component degrades to PM due or down. This group replacement is similar to the control limit policy of a single-unit, but is much more complex. The added complexity is the result of a state space explosion when the number of components and/or the number of degradation levels is large. Because of this state space explosion, the authors do not provide an exact analytical solution. Instead, they propose an approximation method which contracts the state space in order to find a tractable numerical approximation to the minimum expected long-run cost.

The *opportunistic maintenance policy* is closely related to the group maintenance policy. Consider a system in which the cost to perform maintenance on multiple units simultaneously is less than the sum of performing maintenance on each unit individually. In such a case, performing maintenance on one component may create an economically advantageous opportunity to perform maintenance on other components. This generally is the result of some setup cost which is incurred each time maintenance is performed. Most research concerning opportunistic maintenance policies consider a k-out-of-n system. That is, the system operates if and only if at least k out of a total of n components operates. Often, the opportunistic maintenance policy is combined with a variant of a single-unit policy such as a control-limit policy or an age maintenance policy.

The grandfather of opportunistic maintenance policies is the model which considers a system of only two components. One of the earliest research efforts of this type was contributed in 1962 by Radner and Jorgenson [33]. In this work, component 0 and component 1 are independent and in series meaning that the failure of either

component causes the failure of the system. The only type of maintenance action considered for either component is replacement, but the status of component 0 can only be determined at the time that component 1 is being replaced. Component 1 is assumed to have an exponentially distributed lifetime, but component 0 is assumed to have a generally distributed lifetime. The time to replace both components at the same time is less than the time to replace each component individually. The objective is to maximize the expected total discounted time that the system is operating by determining the best maintenance strategy for component 0. The solution method is a dynamic program in which an (n, N) policy is found. This policy is defined by three cases:

1. If component 0 has been operating for less than n discrete time units when component 1 is replaced due to failure, then only replace component 1.
2. If component 0 has been operating between n and N time units when component 1 is replaced due to failure, then replace both components.
3. If component 0 has been operating for more than N time units and no failure has occurred in component 1, then only replace component 0.

The two component problem has been expanded in numerous papers. Radner and Jorgensen [34] extended their previous work [33] by generalizing their model to the case of a finite number of series components in which one component is not monitored and all others are monitored. Vergin and Scriabin [43] considered a two-component and a three-component system in which the components are economically dependent, but stochastically independent. They use dynamic programming to determine an optimal opportunistic preventive maintenance model. A more recent opportunistic model is that of Pham and Wang [29] in which they consider a k-out-of-n:G system[1] with imperfect preventive maintenance and partial failure.

[1] a k-out-of-n:G system is a system consisting of n components, k of which must be good (G) for the system to be operational

2.2.3 Markov Decision Process Maintenance Models

A Markov decision process is the specification of a sequential decision problem for a fully observable environment that satisfies the Markov assumption under additive rewards. These types of problems are often solved via (stochastic) dynamic programming to obtain optimal decision policies. According to Puterman [32:xv], the use of Markov decision processes to solve problems in numerous disciplines first appeared in the 1950's. Of particular interest to this research is the application of Markov decision processes to the maintenance problem. See Section 3.4 for further discussion of Markov decision processes.

Su et al. [38] formulated a Markov decision problem that seeks the optimal preventive maintenance policy over a finite planning horizon for a system that may occupy a finite number of degraded states. There are a finite number of possible maintenance actions that have an associated risk of leaving the system in a worse condition than that prior to the maintenance action. In order to solve the problem, they assume that the planning horizon, T, is much larger than the time between inspections, t. In doing so, they approximate the finite-horizon problem by solving an infinite horizon problem, finding a stationary policy that dictates which action should be taken whenever the system is observed to be in each state, independent of the decision epoch.

Sumter [39] formulated a Markov decision process to find an optimal replacement policy for a satellite constellation over a finite time horizon. The satellite states are assumed to be binary with exponential lifetimes; therefore, the policies found with this model are reactive to failures (i.e., no preventive or corrective maintenance is performed). The objective was to minimize the expected total cost of maintaining a satellite constellation. The actions include replacing a satellite and/or purchasing a satellite to use as a spare, or do nothing. Because the purchasing of spares are considered, the policy produced with this model falls under the joint maintenance and inventory class of policies.

Moustafa et al. [23] allowed continuous inspections and non-constant failure rates by formulating a semi-Markov decision process to find an optimal maintenance policy for a multi-state semi-Markovian deteriorating system. In this paper, the objective was to minimize the long-run cost rate while performing continuous inspections with the actions being do nothing, perform minimal repairs, or replace the system. An optimal control-limit policy can be obtained using the policy-iteration algorithm described by Puterman [32:174].

2.3 Research Contributions

Current satellite constellation maintenance problems are strictly viewed as replacement problems. As such, the tools developed to help determine constellation maintenance policies are aimed at finding the best time to acquire and launch a new satellite. Various performance measures are used to help determine which constellations are in need of a replacement satellite in order to set priorities in a resource allocation problem. These methods of determining satellite constellation maintenance strategies may result in the unnecessary expenditure of hundreds of millions of dollars for a satellite that is not needed.

Optimal maintenance policies have been studied extensively since the industrial revolution. Most optimal maintenance models seek to determine a policy in which the system will receive preventive maintenance to extend the life of the system and avoid penalties incurred due to system failures. Policies of this type generally rely on the assumption that systems have an increasing failure rate. However, it has been argued that complex systems have historically exhibited constant failure rates (see for example [4:18]). There is some optimal maintenance policy research in the literature which addresses systems with constant failure rates such as [13] and [33]; however, these models generally consist of identical units or are combined with units that do not have constant failure rates.

No models could be found in the literature concerning the optimal maintenance of multi-unit system in which each component may exhibit a unique constant failure rate. This thesis will narrow this gap in the literature by presenting a methodology to determine an optimal maintenance policy for a multi-unit system with specific application to a satellite constellation. A constellation is a multi-unit system composed of multi-component satellites. Each satellite component is assumed to possess a constant failure rate, however, the components are not necessarily identical. The optimal policy will be obtained by modelling the entire process as a Markov decision process as described in Chapter 3.

3. Formal Model Description

This chapter presents a formal mathematical model to determine an optimal maintenance policy for degrading satellite constellations. Section 3.1 provides definitions used throughout and describes the system under consideration. Section 3.2 describes the model assumptions employed to facilitate the mathematical model. Section 3.3 presents a stochastic model for a satellite constellation's degradation process as a discrete-time Markov chain (DTMC). Section 3.4 presents the parameters of an optimization problem that can be solved to find an optimal maintenance policy over a finite planning horizon while Section 3.5 formulates the optimization problem as a stochastic dynamic program. Finally, Section 3.6 provides a description of the evaluation methodology used to solve the stochastic dynamic program.

3.1 System Definitions

Previous work has considered the status of satellites to be binary, either up or down (e.g., Sumter [39]). This research considers satellites that may be degraded but remain partially capable. In order to describe a constellation with partially capable satellites, the following definitions are needed.

Definition 3.1 *A satellite function is an operation that must be performed for a satellite to successfully carry out its mission.*

Definition 3.2 *A satellite function is operational if it operates as originally intended.*

Definition 3.3 *A satellite function is non-operational if it does not operate as originally intended.*

Consider a satellite constellation composed of K satellites each having a countable number of degradation levels. Satellite k has a total of $M^{(k)}$ functions that must

be operational to be considered fully capable. If satellite k has fewer than $M^{(k)}$ operational functions, then the satellite is considered to be degraded. The satellites within the constellation are not necessarily identical, the number of functions or the collections of functions may be different. The satellites are assumed to be stochastically independently of one another; however, the cost structure is such that the satellites are economically dependent.

The satellite constellation is inspected at discrete time increments (e.g. daily, weekly, or monthly). The inspections are perfect, meaning that the true state of the constellation is observed at each inspection epoch with probability 1. Each time the constellation is observed, a maintenance action is chosen based on the observed degradation level of the satellite. In this context, maintenance actions consist of doing nothing, performing on-orbit repairs, or performing a satellite replacement.

3.2 Model Assumptions

In order to represent the temporal evolution of the degradation of the constellation, it is necessary to make a few simplifying assumptions. These assumptions aide in reducing the complexity of the problem thereby ensuring mathematical tractability. The first set of assumptions are with regard to *individual* satellites:

- The lifetime distributions of satellite functions are mutually independent. This means that the failure of one function does not contribute to the failure of any other function.
- Satellite functions fail at a constant rate. This implies that the lifetime of each satellite function is exponentially distributed. Satellite functions are very complex, having many individual components that can fail and cause the function to fail. Also, these functions are expected to operate for long periods of time. Some communication satellites are expected to operate for at least 10 years [41]. As noted in Barlow and Proschan [4:18], the lifetimes of com-

plex machines operating for long periods tend to be distributed exponentially. The impact of this assumption is that the optimal maintenance policy will not contain any preventive maintenance (maintenance on an operational function) because preventive maintenance will not decrease the chance that the function becomes non-operational in the next instant.

The second set of assumptions are with regard to satellite *constellations*:

- Each satellite in the constellation is subject to degradation independent of all other satellites. This means that degradation of one satellite does not contribute to the degradation of any other satellite in the constellation.
- If two or more satellites have the same function, the failure rate of the function is equal for each satellite.

The third set of assumptions are made with regard to *on-orbit repairs*:

- If a satellite is repaired on-orbit, it is assumed that the repair is performed instantaneously. (In reality, an on-orbit repair may require months to initiate and complete.)
- There is a positive probability that a repair attempt will be unsuccessful. This implies that a repair attempt has an associated level of risk (i.e., an attempted satellite repair may not improve the status of the satellite).
- If a satellite function is repaired successfully, the function is assumed to remain operational until the next inspection epoch (i.e., the function will not fail in the same time period in which it is repaired).
- If the satellite function is repaired successfully, it is assumed that it is repaired to "good as new" condition.
- On-orbit repair attempts may only be made on functions that are non-operational. Furthermore, a repair attempt will not cause the degradation of any operational functions.

- If the decision is made to perform an on-orbit repair, then every non-operational function on the satellite is repaired.

The fourth set of assumptions are made with regard to satellite *replacements*:

- If a satellite is replaced, it is assumed that the replacement is performed instantaneously. In reality, a satellite may require months to construct and launch into orbit.
- If a satellite is replaced, it is replaced with an identical satellite.
- There is a positive probability that a function will not survive a satellite replacement attempt. This implies that a replacement attempt also has an associated level of risk. The extreme forces applied to a satellite during launch, as well as the harsh nature of the space environment, are assumed to induce a positive probability that a function will arrive on-orbit non-operational.
- If a satellite function arrives on-orbit operational, then the function is assumed to remain operational until the next inspection epoch (i.e., a function that survives a satellite replacement attempt will not degrade in the same time period as the replacement attempt).

3.3 Stochastic Degradation Model

This section presents the mathematical characterization of a satellite constellation's degradation process. In order to determine the degradation level of a constellation, it is necessary to consider the factors that contribute to the overall likelihood of success for the intended mission. Reeves [35:317] explains that mission success criteria can be described as a list that contains all of the events and operations that must occur for the mission to be successful. For the satellite constellation degradation model, the operations necessary for mission success are the satellite functions. The stochastic degradation model is developed by first mathematically characterizing the

degradation process that each individual satellite undergoes and then extending the model to a multiple satellite constellation.

3.3.1 Individual Satellite Degradation Process

A satellite may be required to perform several functions in order to successfully perform its intended mission. Just as any other complex machine, its functions are subject to failure due to degradation over time. This degradation process can be modelled stochastically as a time-homogeneous discrete-time Markov chain (DTMC) requiring the following assumptions (in addition to those listed in Section 3.2):

1. There is a discrete, finite number of degradation levels for each satellite.
2. The probability that the satellite will transition between various degradation levels between any two inspection epochs is the same for all inspection epochs. This property is known as *time homogeneity*.
3. The history of the stochastic degradation process is captured in the current degradation level. In other words, the probability that the satellite will transition between various degradation levels only depends on the current degradation level irrespective of the past.

Each function's operational status evolves stochastically over time. For satellite k at inspection epoch n, the status of function m is a binary random variable defined by

$$X_m^{(k)}(n) = \begin{cases} 1 & \text{if the } m^{th} \text{ function of satellite } k \text{ is operational at time } n \\ 0 & \text{if the } m^{th} \text{ function of satellite } k \text{ is non} - \text{operational at time } n \end{cases} . \tag{3.1}$$

For example, $X_1^{(3)}(2) = 1$ indicates that at inspection epoch 2, function 1 of satellite 3 is observed to be operational.

Because the state of each function is binary, the transition probabilities can be modelled by reliability functions. In other words, the probability that a function

survives one time step is the interval reliability of the subsystems of which it is composed. Define R_m and F_m as the *interval reliability function* and the *interval lifetime distribution* of satellite function m, respectively. The interval lifetime distribution provides the probabilities that the satellite function will become non-operational during the next time interval Δt given that it has survived to the current time t. Recall from Section 3.2 that the lifetime of each function is distributed exponentially. If we let λ_m^{-1} be the mean lifetime of function m, then the lifetime of satellite function m is distributed exponentially with rate parameter λ_m.

Proposition 3.1 *The interval lifetime distribution F_m does not depend on the time of inspection. That is*

$$F_m(t + \Delta t|t) = 1 - e^{-\lambda_m \Delta t}. \tag{3.2}$$

Proof. Define $F_m(t + \Delta t|t)$ as the probability that function m fails sometime in the interval $(t, t + \Delta t)$ given that it has survived to time t. In other words, if the random lifetime of function m is defined as Y_m then

$$F_m(t + \Delta t|t) = P\{Y_m < t + \Delta t | Y_m > t\}. \tag{3.3}$$

After unconditioning, and taking the compliment, Equation (3.3) becomes

$$1 - F_m(t + \Delta t|t) = \frac{P\{Y_m > t + \Delta t,\ Y_m > t\}}{P\{Y_m > t\}}. \tag{3.4}$$

The joint probability in the numerator of Equation (3.4) is captured in the probability of the single event $Y_m > t + \Delta t$ yielding

$$1 - F_m(t + \Delta t|t) = \frac{P\{Y_m > t + \Delta t\}}{P\{Y_m > t\}}. \tag{3.5}$$

Because the lifetimes are assumed to be exponentially distributed, Equation (3.5) is equivalent to

$$1 - F_m(t + \Delta t|t) = \frac{e^{-\lambda_m(t+\Delta t)}}{e^{-\lambda_m t}}. \tag{3.6}$$

where λ_m is the failure rate (failures per period) of function m. Simplifying Equation (3.6) yields

$$\begin{aligned} 1 - F_m(t + \Delta t|t) &= \frac{e^{-\lambda_m t} e^{-\lambda_m \Delta t}}{e^{-\lambda_m t}} \\ &= e^{-\lambda_m \Delta t}. \end{aligned} \tag{3.7}$$

Therefore, the interval failure distribution is given by

$$F_m(t + \Delta t|t) = 1 - e^{-\lambda_m \Delta t} \tag{3.8}$$

which is independent of the current time t. ■

In the satellite maintenance model, Δt corresponds to the time between decision epochs which is equal to a single time step. Therefore,

$$F_m(t + \Delta t|t) = F_m(t + 1|t).$$

If we let $F_m \equiv F_m(t + 1|t)$, then

$$F_m = 1 - e^{-\lambda_m} \tag{3.9}$$

which depends only on the failure rate of function m.

Using Proposition 3.1, the interval reliability can be easily derived. By definition, the reliability is the complement of the lifetime distribution. Therefore,

$$\begin{aligned} R_m &= 1 - F_m \\ &= 1 - (1 - e^{-\lambda_m}) \\ &= e^{-\lambda_m}. \end{aligned} \tag{3.10}$$

Each satellite's degradation level is a row vector of each of its functions' states. Mathematically, the state of satellite k having $M^{(k)}$ functions at inspection epoch n is given by

$$\mathbf{X}^{(k)}(n) = (X_1^{(k)}(n), X_2^{(k)}(n), \ldots, X_{M^{(k)}}^{(k)}(n)). \tag{3.11}$$

In other words, $\mathbf{X}^{(k)}(n)$ describes the random state of satellite k at the n^{th} time step as the vector of each function's states. For example, if $M^{(1)} = 3$, then the observation of $\mathbf{X}^{(1)}(5) = (1, 1, 0)$ indicates that at inspection epoch 5, satellite 1 has functions 1 and 2 operational and function 3 non-operational. This is an example of a partially capable satellite.

Since $\mathbf{X}^{(k)}(n)$ evolves randomly over time, the degradation process of satellite k is given by the discrete-time stochastic process

$$\{\mathbf{X}^{(k)}(n) : n \geq 0\}.$$

For the remainder of the single satellite formulation the superscript k is dropped because there is only one satellite in the constellation being modelled.

A satellite's level of degradation evolves over time and is characterized by the stochastic process $\{\mathbf{X}(n) : n \geq 0\}$. Associated with the stochastic process $\{\mathbf{X}(n) : n \geq 0\}$ are a governing set of transition probabilities denoted as $p_{\mathbf{i},\mathbf{j}}$. The transition probability, $p_{\mathbf{i},\mathbf{j}}$, is the probability that the satellite will degrade to level $\mathbf{j}$ given its history of degradation and current state $\mathbf{i}$ by the next inspection epoch

where $\mathbf{i}$ and $\mathbf{j}$ are realizations of the random vectors $\mathbf{X}(n)$ and $\mathbf{X}(n+1)$, respectively. This transition probability can be written as

$$p_{\mathbf{i},\mathbf{j}} = P\{\mathbf{X}(n+1) = \mathbf{j} | \mathbf{X}(n) = \mathbf{i}, \mathbf{X}(n-1), \ldots, \mathbf{X}(0)\}. \tag{3.12}$$

A satellite transition event is composed of the intersection of each of its functions transition events. Therefore, the transition probabilities for each satellite's degradation process are the intersection of the transition probabilities for each of its functions.

By Proposition 3.1, Equation (3.12) can be simplified because each function's transition probabilities are dependent only on the interval lifetime distribution and the current state of the function. Furthermore, the interval lifetime distribution is only dependent on the length of the interval Δt which is 1. Therefore, the history of the process is captured in the current state of the satellite. Knowledge of the state of the system prior to the current state is not necessary to compute the satellite transition probabilities. Equation (3.12) simplifies to

$$p_{\mathbf{i},\mathbf{j}} = P\{\mathbf{X}(n+1) = \mathbf{j} | \mathbf{X}(n) = \mathbf{i}\}. \tag{3.13}$$

This is known as the *Markov*, or memoryless property.

Because the transition probabilities are assumed to be time-homogeneous, Equation (3.13) is equivalent to

$$p_{\mathbf{i},\mathbf{j}} = P\{\mathbf{X}(1) = \mathbf{j} | \mathbf{X}(0) = \mathbf{i}\}. \tag{3.14}$$

By the stated assumptions and Equations (3.13) and (3.14), we can state that the stochastic process $\{\mathbf{X}(n) : n \geq 0\}$ is a time-homogeneous DTMC with transition probability matrix $\mathbf{P} = [p_{\mathbf{i},\mathbf{j}}]$.

In the case of the single satellite model, the transition probability matrix of satellite k is a $2^{M^{(k)}} \times 2^{M^{(k)}}$ matrix. When the states are indexed properly, the transition probability matrix can either be upper or lower triangular.

3.3.2 Constellation Degradation Process

In order to account for multiple satellites the superscript, k, is returned to the state description of each individual satellite. In other words, the stochastic process that characterizes the degradation process of satellite k is given by

$$\{\mathbf{X}^{(k)}(n) : n \geq 0\} \equiv \{(X_1^{(k)}(n), X_2^{(k)}(n), \ldots, X_M^{(k)}(n)) : n \geq 0\}. \tag{3.15}$$

Each satellite in a constellation is subject to the degradation process described in Section 3.3.1. As indicated by the assumption that satellites in the constellation are not necessarily identical, each satellite in the constellation may have a different number of elements in its state vector. As an example, take a constellation with two satellites. Satellite 1 has two functions and Satellite 2 has both functions of satellite 1, plus an additional function. In this case, $K = 2$, $M^{(1)} = 2$, and $M^{(2)} = 3$. Therefore, the random row vector $\mathbf{X}^{(1)}(n)$ would have two components and the random row vector $\mathbf{X}^{(2)}(n)$ would have three components. Furthermore,

$$\mathbf{X}^{(1)}(n) \equiv (\mathbf{X}_1^{(1)}(n), \mathbf{X}_2^{(1)}(n))$$

and

$$\mathbf{X}^{(2)}(n) \equiv (\mathbf{X}_1^{(2)}(n), \mathbf{X}_2^{(2)}(n), \mathbf{X}_3^{(2)}(n)).$$

The multiple satellite model closely resembles the single satellite model, however, the constellation model has a significantly larger state space. The overall state of the constellation is a row vector of the states of each of the satellites within the constellation. If there are K satellites in the constellation, then at inspection epoch

n the state of the constellation is given by

$$\mathbf{X}(n) = (\mathbf{X}^{(1)}(n), \mathbf{X}^{(2)}(n), \ldots, \mathbf{X}^{(K)}(n)). \tag{3.16}$$

Using the previous example, the observation

$$\mathbf{X}(5) = ((1, 1), (0, 1, 1))$$

indicates that at inspection epoch 5, satellite 1 is fully capable and satellite 2 is partially capable.

Since $\mathbf{X}(n)$ evolves stochastically over time, the overall degradation process of the constellation is given by the discrete time stochastic process

$$\{\mathbf{X}(n) : n \geq 0\}.$$

Furthermore, because each satellite's degradation process is a time-homogeneous DTMC and each satellite is stochastically independent, the constellation degradation process $\{\mathbf{X}(n) : n \geq 0\}$ is also a time-homogeneous DTMC.

The transition probabilities for the multiple satellite model are similar to those in the single satellite model. However, the transition probability matrix is much larger due to the additional states. In the case of the multiple satellite model, the transition probability matrix is a $2^{\sum_{k=1}^{K} M^{(k)}} \times 2^{\sum_{k=1}^{K} M^{(k)}}$ matrix. For example, if a constellation consists of two satellites with $M^{(1)} = 2$ and $M^{(2)} = 3$ then the transition probability matrix contains $2^{2+3} = 32$ rows and columns. As in the individual satellite formulation, when the states are indexed properly, the transition probability matrix can be either upper or lower triangular.

A constellation transition event is composed of the intersection of each of its satellites transition events. Therefore, the transition probabilities for the constellation degradation process are the intersection of the transition probabilities for each of

its satellites. But since the satellites are stochastically independent, the intersection can be expressed as a product.

The case of a one satellite constellation, $K = 1$, is a special case of the constellation formulation and the model becomes identical to that of section 3.3.1. As an illustration of this, take a one satellite constellation in which the satellite has two functions. Then the constellation's random state at inspection epoch n is given by

$$\mathbf{X}(n) \equiv \mathbf{X}^{(1)}(n).$$

But

$$\mathbf{X}^{(1)}(n) = (X_1^{(1)}(n), X_2^{(1)}(n)).$$

Therefore,

$$\mathbf{X}(n) \equiv (X_1^{(1)}(n), X_2^{(1)}(n))$$

which is identical to the single satellite formulation.

To illustrate the stochastic degradation process, consider a two satellite constellation in which Satellite 1 has 2 functions and Satellite 2 has 1 function which is different than either function of Satellite 1. In other words, $K = 2$, $M^{(1)} = 2$, and $M^{(2)} = 1$. Therefore, there are $2^{2+1} = 8$ total degradation levels. The enumerated degradation levels for this constellation are given in Table 3.1.

Table 3.1 Two satellite state space.

State	*Degradation Level*
s_1	$((1,1),(1))$
s_2	$((1,1),(0))$
s_3	$((1,0),(1))$
s_4	$((1,0),(0))$
s_5	$((0,1),(1))$
s_6	$((0,1),(0))$
s_7	$((0,0),(1))$
s_8	$((0,0),(0))$

A possible sample path of $\{\mathbf{X}(n) : n \geq 0\}$ is given in Figure 3.1. When no maintenance actions are considered, the constellation cannot transition to a less degraded state as indicated by the non-increasing sample path.

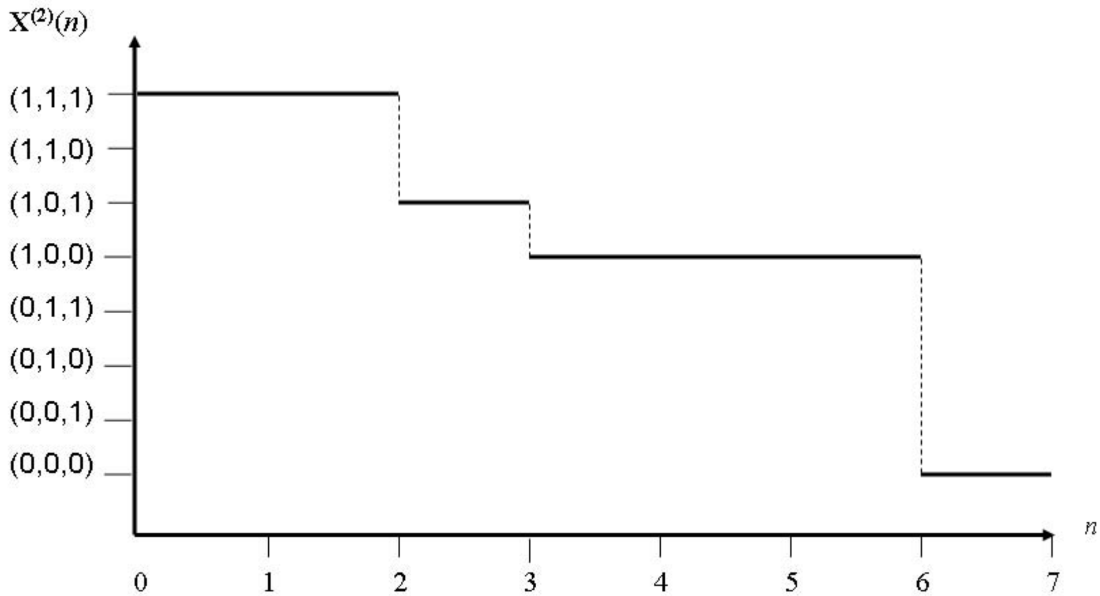

Figure 3.1 Possible sample path for $\{\mathbf{X}(n) : n \geq 0\}$

The transition probability matrix is an 8×8 matrix. There are three different functions in the constellation, therefore, there are three different values for each of R_m and F_m. For function 1 of satellite 1, $m = 1$. For function 2 of satellite 1, $m = 2$. For the function on satellite 2, $m = 3$. The transition probability matrix is easily

determined to be

$$\mathbf{P} = \begin{bmatrix} R_1R_2R_3 & R_1R_2F_3 & R_1F_2R_3 & R_1F_2F_3 & F_1R_2R_3 & F_1R_2F_3 & F_1F_2R_3 & F_1F_2F_3 \\ 0 & R_1R_2 & 0 & R_1F_2 & 0 & F_1R_2 & 0 & F_1F_2 \\ 0 & 0 & R_1R_3 & R_1F_3 & 0 & 0 & F_1R_3 & F_1F_3 \\ 0 & 0 & 0 & R_1 & 0 & 0 & 0 & F_1 \\ 0 & 0 & 0 & 0 & R_2R_3 & R_2F_3 & F_2R_3 & F_2F_3 \\ 0 & 0 & 0 & 0 & 0 & R_2 & 0 & F_2 \\ 0 & 0 & 0 & 0 & 0 & 0 & R_3 & F_3 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

3.4 Optimization Problem

Now that the degradation process for a satellite constellation has been described, we can formulate an optimization problem that can be solved to find an optimal maintenance policy for the constellation. The first step in formulating an optimization problem is to define the goal of optimization problem in terms of an objective function. The second step is to determine the decision variables, the variables that can be changed to affect the objective function. The third step is to identify any constraints imposed on the values of the decision variables. Finally, a suitable optimization technique must be chosen.

3.4.1 Objective Function

The goal of the optimization problem is to find the best maintenance policy over a finite planning horizon. In order to determine which policy is "best," a value must be assigned to each policy so that they can be represented in an ordered set. The objective function defines how values are assigned to each policy and determines whether to seek the maximum or the minimum of this set (assuming a finite number of policies). Some of the candidate objectives considered in this research were

- minimizing the total expected cost of maintaining the constellation,
- maximizing mission effectiveness,
- maximizing profit,
- minimizing loss of service, and
- minimizing the total loss of utility.

After careful consideration, the objective of minimizing the expected total loss of utility was chosen. This objective was chosen because there are two types of costs associated with maintaining a satellite constellation. The first cost is a deterministic monetary cost incurred when a maintenance action is chosen. For example, when a satellite is replaced, there is a known procurement cost. The second cost is a penalty cost, possibly monetary in the private sector, or some other unit of mission effectiveness in the military sector. For example, when a commercial communication satellite is operating in a partially capable state, a monetary value associated with loss of customer goodwill might be incurred; but in the military sector, the cost might be lost lives because of a communications error on the battlefield. When considering the business application, the utility is measured on the real line in units of dollars. When considering the military application, the two types of cost are in different units and must therefore be transformed to some common unit of utility.

3.4.2 Decision Variables

The decision variables for this problem are the maintenance actions to take at decision epochs when the constellation is observed to be in a certain state. In other words, which maintenance action should be taken when the constellation is observed to be in state s at time n for every possible s and n?

The feasible maintenance actions for this model are do nothing, perform one or more on-orbit repairs, or replace one or more satellites. Previous constellation maintenance models have considered only replacements as a feasible means of repairs

due to the high cost involved with performing an on-orbit repair (e.g. Sumter [39]). However, on-orbit repairs may become a more viable option with the development of satellite repair robots which could repair satellites at a fraction of the cost of a manned mission [27].

3.4.3 Solution Strategy

We formulate the problem as a Markov decision process and solve it via stochastic dynamic programming. This formulation was chosen because the maintenance policy it produces can be mathematically proven to be optimal. An additional reason that this technique was selected is that it produces a policy that is easy to interpret. An illustration of the ease of interpretability is provided in Chapter 4.

According to Puterman [32:17], "a Markov decision process model consists of five elements: decision epochs, states, actions, transition probabilities, and rewards." A clarification of each of the five elements of a MDP is provided below.

Definition 3.4 *A decision epoch is a point in time at which the system is inspected and a decision is made to affect the system based on that observation. The set of decision epochs can be finite or infinite; discrete or a continuum.*

Definition 3.5 *At each decision epoch, the state of the system is what is observed and an allowable action is chosen based on that observed state. According to Puterman [32:18] the state and action sets "may each either be*

1. *arbitrary finite sets;*
2. *arbitrary countably infinite sets;*
3. *compact subsets of finite dimensional Euclidean space; or*
4. *non-empty Borel subsets of complete, separable metric spaces."*

Definition 3.6 *Each time an action is chosen, the system evolves according to a probability distribution determined by the observed state of the system and the*

action chosen. The collection of these probability distributions is known as the transition probabilities. This set also includes the effect transitions, or the satellite degradation transition matrix.

Definition 3.7 *A reward is received each time an action is chosen. The reward takes on positive values for "income" or "gain", and negative values for "cost" or "loss". According to Puterman [32:20], the "reward might be*

1. *a lump sum received at a fixed or random time prior to the next decision epoch;*
2. *accrued continuously throughout the current period;*
3. *a random quantity that depends on the system state at the subsequent decision epoch; or*
4. *a combination of the above."*

In this application, a MDP is formulated to determine a policy that minimizes the loss associated with maintaining a degrading satellite constellation. A *policy* is denoted by a vector $\pi = (d_1, d_2, \ldots, d_{N-1})$ such that d_n indicates the decision rule for decision epoch n. The *decision rule* for decision epoch n provides an action to take for each state in which the system may be observed at time n. (Note that there is no decision made at the final epoch, N.) An optimal policy is denoted by $\pi^* = (d_1^*, d_2^*, \ldots, d_{N-1}^*)$ where d_n^* indicates the optimal decision rule for decision epoch n. In other words, for each n, d_n^* is the vector of optimal actions to take for every possible state in which the system could be observed at decision epoch n. For example, if $d_3^* = (1, 1, 4)$ then, if at decision epoch 3 the system is observed to be in states 1 or 2, action 1 is optimal; if at decision epoch 3 the system is observed to be in state 3, action 4 is optimal.

To illustrate the policy produced by a Markov decision process, consider a single satellite with two functions. The planning horizon is 10 periods and the allowable actions are:

1. - do nothing,

2. - repair the satellite, or

3. - replace the satellite.

Then an optimal policy may be as given in Table 3.2. If the current epoch is $n = 3$, then the optimal decision rule is $d_3 = (1, 3, 2, 3)$. This means that at time $n = 3$, if the satellite is observed to be in state s_1, then the optimal action to take is to do nothing. If the satellite is observed to be in states s_2 or s_4 then the optimal action is to replace the satellite. If the satellite is observed to be in state s_3 then the optimal action is to perform an on-orbit repair.

Table 3.2 Optimal policy example.

State	Decision Epoch (n)									
	1	2	3	4	5	6	7	8	9	10
$s_1 = (1,1)$	1	1	1	1	1	1	1	1	1	N/A
$s_2 = (1,0)$	3	3	3	1	1	1	1	1	1	N/A
$s_3 = (0,1)$	2	2	2	2	2	2	1	1	1	N/A
$s_4 = (0,0)$	3	3	3	3	3	3	3	3	3	N/A

3.5 Formal Markov Decision Process Formulation

In formulating the satellite maintenance model as a Markov decision process, we assume that the action sets are *stationary*. This means that the set of allowable actions at decision epoch 1 is the same as the set of allowable actions at decision epoch n for every n. The elements of a MDP with regard to the satellite maintenance model follow.

3.5.1 Decision Epochs

In the case of the infinite horizon problem, a stationary policy only exists if the underlying Markov chain is ergodic (irreducible, recurrent, and aperiodic). This condition is not guaranteed to hold for the satellite maintenance model. When

considering a finite planning horizon, at least one optimal policy will always exist. Therefore, the planning horizon for the satellite constellation maintenance model is finite. Also, inspections of a constellation occur at constant intervals to determine its operational status. Therefore, the time steps are discrete. The set of decision epochs is given by

$$\mathcal{N} = \{1, 2, \ldots, N\},\ N < \infty. \tag{3.17}$$

In other words, the inspections and decisions are made at discrete intervals. They could represent inspections that occur daily, quarterly, annually, etc.

3.5.2 State Space

The state space for each individual satellite in the MDP model is composed of a finite number of degradation levels as described in section 3.3.1. The random state of satellite k of the constellation is given by

$$\mathbf{X}^{(k)}(n) \equiv (X_1^{(k)}(n), X_2^{(k)}(n), \ldots, X_M^{(k)}(n)). \tag{3.18}$$

The state space for the entire constellation is composed of a finite number of degradation levels as described in Section 3.3.2. The random state of the constellation at decision epoch n can be written as

$$\mathbf{X}(n) \equiv (\mathbf{X}^{(1)}(n), \mathbf{X}^{(2)}(n), \ldots, \mathbf{X}^{(K)}(n)) \tag{3.19}$$

where $\mathbf{X}^{(k)}(n)$ is the random state of satellite k at inspection epoch n.

The total number of states, or the *cardinality* of satellite k's state space $\mathcal{S}_k$, is

$$|\mathcal{S}_k| = 2^{M^{(k)}} \tag{3.20}$$

and the cardinality of the state space for the entire constellation, $\mathcal{S}$, is

$$|\mathcal{S}| = 2^{\sum_{k=1}^{K} M^{(k)}}. \tag{3.21}$$

3.5.3 Actions

Satellite k has an associated action set denoted by $\mathcal{A}^{(k)}$ that is an indexed listing of all possible actions that can be taken for satellite k. Some actions in this set may not be allowed when satellite k is observed to be in certain states. Therefore, the action set for every state s is a subset of $\mathcal{A}^{(k)}$ denoted by $\mathcal{A}_s^{(k)}$. These satellite action subsets are the same for every decision epoch due to the assumption of stationary action sets. If s denotes the observed state of satellite k and i denotes the index of the action, then a specific action for satellite k in state s is given by

$$a_{s,i}^{(k)} \in \mathcal{A}_s^{(k)}.$$

For example, action 2 for satellite 5 when observed to be in state s_1 is denoted by $a_{s_1,2}^{(5)}$.

The action set for the entire constellation, denoted by $\mathcal{A}$, can be described as an indexed listing of all possible combinations of individual satellite actions. As in the individual satellite action sets, not all actions are allowable for every state the constellation can occupy. Therefore, the action set for every constellation state s' is a subset of $\mathcal{A}$ denoted by $\mathcal{A}_{s'}$. If s' denotes the observed state of the constellation and i' denotes the index of the constellation action, then a specific action for the constellation in state s' is given by the K-dimensional row vector

$$\mathbf{a}_{s',i'} \in \mathcal{A}_s = \{(a_{s,i}^{(1)}, \ldots, a_{s,i}^{(K)}) : a_{s,i}^{(k)} \in \mathcal{A}_s^{(k)}\} \tag{3.22}$$

As an illustration, take a constellation that consists of two satellites with two possible actions for each satellite, then

$$\mathbf{a}_{s'_9,2'} = (a^{(1)}_{s_3,1}, a^{(2)}_{s_1,2})$$

would indicate that constellation action $2'$ when the constellation is in state s'_9 means to take action 1 for the first satellite when in that satellite's state s_3 and action 2 for the second satellite when in that satellite's state s_1.

3.5.4 Transition Probabilities

The following parameters are necessary to determine the transition probabilities:

- R_m and F_m are respectively the interval reliability and the interval lifetime distribution of satellite function m. The lifetime of satellite function m is assumed to be distributed exponentially with rate parameter λ_m.
- H_m is the Bernoulli probability mass function (p.m.f.) associated with the probability that function m is successfully repaired.
- G_m is the Bernoulli p.m.f. associated with the probability that function m survives a satellite replacement attempt.

As in Su et al. (2000) [38], H_m and G_m imply the possibility of *imperfect maintenance*. This means that a measure of risk is associated with choosing a maintenance action (i.e., there is a possibility that the system will not improve as a result of performing the maintenance action).

In order for the constellation to transition from one state to another, each satellite in the constellation must undergo a transition, even if the transition is from one state to the same state. Therefore, the transitions of each individual satellite are discussed first followed by a discussion of the transitions of the entire constellation.

Let $h(n)$ denote the history of the states and actions for an individual satellite up to $n-1$. Mathematically, the history is given by

$$h(n) = \{(s(j), a(j)) : 1 \leq j \leq n-1\}$$

where each $s(j)$ is a realization of the random vector $\mathbf{X}^{(k)}(j)$. Denote $p_n^{(k)}\{j|s(n), a(n), h(n)\}$ as the probability at decision epoch n that satellite k will transition to state $j \in \mathcal{S}^{(k)}$ given its current state $s(n)$, the action taken $a(n)$, and the history $h(n)$. For example

$$p_3^{(2)}\{s_1(4)|s_4(3), a_3(3), (s_1(2), a_1(2), s_3(1), a_1(1))\}$$

is the probability at decision epoch 3 that satellite 2 will transition to state s_1 given the current state s_4, the action taken a_3, and the history $h(3)$ (state s_1 and action a_1 at $n=1$, state s_3 and action a_1 at $n=2$).

By the Markov assumption, the probability at decision epoch $n+1$ that the constellation will transition to state $j \in \mathcal{S}$ given the current state $s(n)$, the current action $a(n)$, and the history $h(n)$, is dependent only on the most recent observed level of degradation and the action taken. That is,

$$p_n^{(k)}\{j|s(n), a(n), h(n)\} = p_n^{(k)}\{j|s(n), a(n)\} \tag{3.23}$$

where $n = 1, 2, \ldots, N$ and $k = 1, 2, \ldots, K$.

As in the stochastic degradation model, the constellation transition events are composed of the intersection of each of its satellites transition events. Therefore, the transition probabilities for the constellation are the intersection of the transition probabilities for each of its satellites. But since the satellites are independent, the intersection can be expressed as a product. Proposition 3.2 clarifies this point.

Proposition 3.2 *Suppose that at decision epoch n, a satellite constellation is observed to be in state $s \in \mathcal{S}$. Then the probability at decision epoch $n+1$ that the*

constellation will transition to state $j \in \mathcal{S}$ given the current state s and the current action vector $\mathbf{a}_{s,i}$ is given by

$$p_n\left\{j|s, \mathbf{a}_{s,i}\right\} = \prod_{k=1}^{K} p_0^{(k)}\left\{j^{(k)}|s^{(k)}, a_{s^{(k)},i}\right\} \tag{3.24}$$

where $s^{(k)}, j^{(k)} \in \mathcal{S}^{(k)}, k = 1, \ldots, K$.

Proof. By definition,

$$p_n\left\{j|s, \mathbf{a}_{s,i}\right\} = p_n\left\{j^{(1)}, j^{(2)}, \ldots, j^{(K)}|s^{(1)}, s^{(2)}, \ldots, s^{(K)}, a_{s^{(1)},i}, a_{s^{(2)},i}, \ldots, a_{s^{(K)},i}\right\}. \tag{3.25}$$

By the assumption that satellites are subject to degradation independently of each other, the joint probability in Equation (3.25) is equivalent to

$$p_n\left\{j|s, \mathbf{a}_{s,i}\right\} = p_n^{(1)}\left\{j^{(1)}|s^{(1)}, a_{s^{(1)},i}\right\} \times p_n^{(2)}\left\{j^{(2)}|s^{(2)}, a_{s^{(2)},i}\right\} \times \ldots \times p_n^{(K)}\left\{j^{(K)}|s^{(K)}, a_{s^{(K)},i}\right\}. \tag{3.26}$$

Since each satellite's degradation process is assumed to have stationary transition probabilities in discrete time, we can state that

$$p_n\left\{j|s, \mathbf{a}_{s,i}\right\} = p_0^{(1)}\left\{j^{(1)}|s^{(1)}, a_{s^{(1)},i}\right\} \times p_0^{(2)}\left\{j^{(2)}|s^{(2)}, a_{s^{(2)},i}\right\} \times \ldots \times p_0^{(K)}\left\{j^{(K)}|s^{(K)}, a_{s^{(K)},i}\right\} \tag{3.27}$$

which when written in product form is the stated hypothesis. ■

3.5.5 Expected Rewards

The expected rewards are necessary to explicitly define the objective function. Each time the constellation is inspected, an action is taken that has a corresponding reward. The rewards in this context are actually utility costs and are, therefore, negative values. Therefore, the rewards will be regarded as losses. The losses in this model are the sum of a deterministic utility cost to perform the action chosen

and the expected penalty cost resulting from that action causing the constellation to transition into any given state over the next time period. The penalty cost can be assigned in a variety of ways. This subsection provides a methodology in which each constellation degradation level is assigned a fixed penalty cost.

There is potentially a different penalty cost associated with each constellation degradation level. Define $C_p(s)$ as the penalty cost associated with constellation state $s \in \mathcal{S}$. The total expected loss at each decision epoch is the immediate cost resulting from performing a maintenance action plus the expected penalty cost as a result of that action over the next time step. If we let $c^{(k)}(a)$ be the cost to perform maintenance action a for satellite k then the reward at time n when taking action a while in state s is given by

$$r_n(s,a) = \sum_{k=1}^{K} c^{(k)}(a) + \sum_{j \in \mathcal{S}} C_p(s_j) p_n(j|s,a) \tag{3.28}$$

where $p_n(j|s,a)$ is determined by Equation (3.24).

Because the planning horizon is finite, there is no decision to make at the final epoch, and the reward function does not apply. Therefore, the reward received at the final decision epoch is the value of the constellation when in state s given by $r_N(s)$. This is known as the *terminal reward* and can be defined by any function of the final states of each satellite.

3.6 MDP Evaluation Methodology

Evaluating a MDP requires that the evaluation criteria be established and the classification of the policy be defined. The evaluation criteria is determined by the goals of the decision maker. Determining the classification of the policy requires knowledge of the evaluation criteria and the properties of the decision rules.

In this research, the optimality criterion is the minimum expected total loss (of utility). This optimality criterion was chosen because the goal of this research is to find a policy that minimizes the total loss of maintaining a satellite over a finite planning horizon. Because the total expected loss is a a total expected reward resulting in a negative value, minimizing the loss is equivalent to maximizing the reward.

A policy's classification is based on the classification of the decision rules of which it is composed. These decision rules are classified by their dependence on the history of the system, and the certainty with which an action is chosen based on that decision rule. A decision rule, d_n, can be deterministic (D) or randomized (R). A deterministic decision rule specifies the actions to take for every given state with certainty while randomized decision rules specify a probability distribution on the set of actions for each state [32:21]. A decision rule, d_n, may also be either history dependent (H) or Markovian (M). A history dependent decision rule determines the actions or probability distribution on the action set for each history of states and actions up to the current decision epoch while a Markovian decision rule determines the actions or probability distribution on the action set for every given state that the system may currently occupy [32:21]. Combining both types of decision rule classifications leads to four possibilities of policy classifications.

$\Pi^{HR} \equiv$ the set of all policies that contain randomized history dependent decision rules.

$\Pi^{HD} \equiv$ the set of all policies that contain deterministic history dependent decision rules.

$\Pi^{MR} \equiv$ the set of all policies that contain Markovian randomized decision rules.

$\Pi^{MD} \equiv$ the set of all policies that contain Markovian deterministic decision rules.

Markov deterministic policies (i.e., those belonging to Π^{MD}) are the easiest to implement and require the least computing power. Also, by the Markov property and the total expected loss optimality criterion, an optimal Markovian deterministic policy $\pi^* \in \Pi^{MD}$ is optimal for all policy classifications [32].

The single satellite MDP has a finite planning horizon with a finite state and action space. Also, the desired optimal policy of the Markov deterministic type. Therefore, the problem can be easily solved using the well-known backward induction (value iteration) method for solving dynamic programs. An example of a backward induction algorithm is given in Puterman [32:92]. The algorithm computes the optimal value of the MDP and the optimal actions that produce the optimal value for each state in each decision epoch. As the name implies, the algorithm computes these values for each decision epoch, beginning at N and working backwards to decision epoch 1 resulting in an optimal policy, $\pi^* \in \Pi^{MD}$.

The backward recursion equations used in the backward induction algorithm to determine the optimal value of the MDP and the optimal actions at decision epoch n for state s are given by

$$v_n^*(s) = \max_{a \in \mathcal{A}_s} \left\{ r_n(s,a) + \sum_{j \in \mathcal{S}} p_n(j|s,a) v_{n+1}^*(j) \right\}, \tag{3.29}$$

and

$$a_s^*(n) = \arg\max_{a \in \mathcal{A}_s} \left\{ r_n(s,a) + \sum_{j \in \mathcal{S}} p_n(j|s,a) v_{n+1}^*(j) \right\} \tag{3.30}$$

respectively with the boundary condition

$$v_N^*(s) = r_N(s) \tag{3.31}$$

for all $s \in \mathcal{S}$. These equations are often referred to as the *Bellman functional equations* (see for example [18]).

The value $v_n^*(s)$ is the optimal value of the MDP from decision epoch n forward when in state s. The value of $a_s^*(n)$ is the action that produces the optimal value of the MDP from decision epoch n forward. In other words, $v_n^*(s)$ is the minimum expected loss incurred throughout the rest of the planning horizon as a result of choosing $a_s^*(n)$ at decision epoch n and in state s.

3.7 Summary

This chapter has provided a methodology for obtaining an optimal maintenance policy for a degrading satellite constellation. First an analytical model for the stochastic degradation process of a satellite constellation was presented. This degradation process was modelled as a time-homogeneous discrete-time Markov chain. Then an optimization problem was formulated that can be solved to find an optimal maintenance policy. The problem was formulated as a stochastic dynamic program, or Markov decision process (MDP). In Chapter 4, the models developed in this chapter are illustrated via two examples. The first case is that of a single satellite constellation, and the second case is a three satellite constellation. Numerical results are computed followed by sensitivity analysis on the model parameters.

4. Numerical Results

In this chapter, we illustrate the model described in Chapter 3 using notional data in two cases. The first case is a one-satellite constellation, and the second case is a three-satellite constellation. A description of the problem data is presented first followed by the explicit formulations and numerical results for each case. Finally we provide a sensitivity analysis on the problem parameters with respect to the optimal policy and optimal value.

4.1 Data

In order to create a realistic notional data set, some typical values for each parameter were chosen by using a number of reliable sources. The required inputs are as follows:

- the expected lifetime of the subsystems that comprise each function,
- the probability of successful repair for each function,
- the cost to perform an on-orbit repair while in state s denoted by $C_r(s)$,
- the probability that each function will survive launch and placement into orbit,
- the cost to replace a satellite denoted by C_s, and
- the penalty costs incurred for functions being non-operational denoted by C_p.

Expected lifetime and replacement cost information were obtained from the Air Force Link Fact Sheets [41] and Jane's Space Directory (2002-2003) [3] with regard to the following satellite constellations:

- Defense Support Program (DSP),
- Defense Meteorological Satellite Program (DMSP),
- Defense Satellite Communications System (DSCS),

- Navstar Global Positioning System (GPS), and
- Milstar Satellite Communications System (Milstar).

On-orbit repair success rates and cost information were gathered from Jane's Space Directory (2002-2003) [3] with regard to the Hubble Space Telescope (HST) servicing missions and launch success rates of the Delta 2 rocket and space shuttles. Additional on-orbit repair cost information was gathered from the On-Orbit Servicing website [27] with regard to satellite repair robots that may be available in the future.

For the purposes of this research, the expected lifetime of a satellite is assumed to be the design life. The design life may have a wide range depending on the mission. Table 4.1 gives the design life values for some satellites in U.S. Air Force maintained constellations.

Table 4.1 Design life of various maintained satellites.

Constellation	*Design Life*
DSP	1.25 years - 5 years
DMSP	1.5 years - 5 years
DSCS	7.5 years - 10 years
GPS	7.5 years - 10 years
Milstar	10 years

Due to high costs, on-orbit repairs are extremely rare; therefore, estimates for the probability that a function will be repaired successfully are rare. For this reason, these probabilities will correspond to the probability of a successful launch for a space vehicle, such as a shuttle or a Delta rocket, and the successful repair rates of the Hubble Space Telescope (HST). According to Jane's Space Directory (2002-2003) [3:265] the Delta 2 rocket has a successful launch rate of 99.01%. Jane's Space Directory (2002-2003) [3:500] also reported that as of February 2001, the shuttle program had similar launch success rates with 101/102 or 99.02% being successful. The HST has had four servicing missions in which each repair was successful, therefore it will

be assumed that the probability of a successful repair given a successful launch will be high (≥ 0.95).

According to Jane's Space Directory (2002-2003) [3], the cost to perform an on-orbit repair to the Hubble Space Telescope in December 1993 was $674 million, $763 million in fiscal year 2000 dollars (FY00$). This included $378 million (FY00$428 million) for the shuttle used to transport the maintenance crew to the Hubble. Other means of performing satellite repairs at a much lower cost are being developed. For example, the On-Orbit Servicing website [27] reported that satellite repair robots are being developed that may potentially perform repairs at an estimated $30 - $50 million.

The event that a satellite is successfully replaced is composed of many subevents. The launch vehicle must successfully carry the satellite to the intended orbit and successfully deploy the satellite. Also, each component on the satellite must survive the trip to orbit and successfully activate when on orbit. The probability of successful launch has already been established to be close to 99%. However, information about individual subsystem survival rates on the satellite could not be found. It is assumed that these probabilities are high (≥ 0.95).

For the purposes of this research, the replacement cost is assumed to be the unit cost of a type of satellite. Table 4.2 gives the unit cost of some typical satellites in United States Air Force maintained constellations as cited in [41] and [3].

Table 4.2 Unit cost of some typical maintained satellites.

Constellation	*Unit Cost (millions)*
DSP	$400
DMSP	$88
DSCS	$200
GPS	$57
Milstar	$800

The penalty costs can be measured as a monetary loss in some commercial applications. However, in military applications, the penalty costs cannot necessarily be measured as a monetary loss. Realistically, in the military application a measure of utility should be assigned to relate the loss of mission effectiveness to the cost of a repair or replacement. For the purposes of this analysis, the penalty costs will be assumed to be in lost dollars to ensure additivity of the loss function.

4.2 One-Satellite Example

In the first example, consider a satellite that must perform three functions to be fully capable. We suppose a decision maker desires an optimal maintenance policy for this satellite over the next 5 years. The satellite will be inspected every quarter, and a decision will be made to either do nothing, repair, or replace the satellite. The constellation contains one satellite ($K = 1$) with three functions ($M = 3$); therefore, there are eight possible degradation levels ($|\mathcal{S}| = 8$). Since four inspections will be done every year for five years there are a total of 20 decision epochs ($N = 20$).

4.2.1 One-Satellite Degradation Process

In the case of a one-satellite constellation, the state of the constellation is equivalent to the state of the individual satellite. Therefore, the random state of the satellite at inspection epoch n is

$$\mathbf{X}(n) = (X_1(n), X_2(n), X_3(n)). \tag{4.1}$$

Because there are three functions that can each have two possible states, the set of all possible states for the satellite has cardinality equal to $2^3 = 8$. The eight possible satellite states are given in Table 4.3.

Because no maintenance actions are being considered at this point, the satellite may only transition into an equivalent state or a more degraded state. Figure

Table 4.3 Enumerated state space for example 1.

State	*Vector Representation*	*State Description*
s_1	$(1,1,1)$	fully capable
s_2	$(1,1,0)$	functions 1 and 2 operational
s_3	$(1,0,1)$	functions 1 and 3 operational
s_4	$(1,0,0)$	function 1 operational
s_5	$(0,1,1)$	functions 2 and 3 operational
s_6	$(0,1,0)$	function 2 operational
s_7	$(0,0,1)$	function 3 operational
s_8	$(0,0,0)$	fully degraded

4.1 gives a sample path that demonstrates the non-increasing nature of the degradation process when no maintenance is performed. This has the effect of making the transition probability matrix upper triangular as can be seen in Equation (4.2).

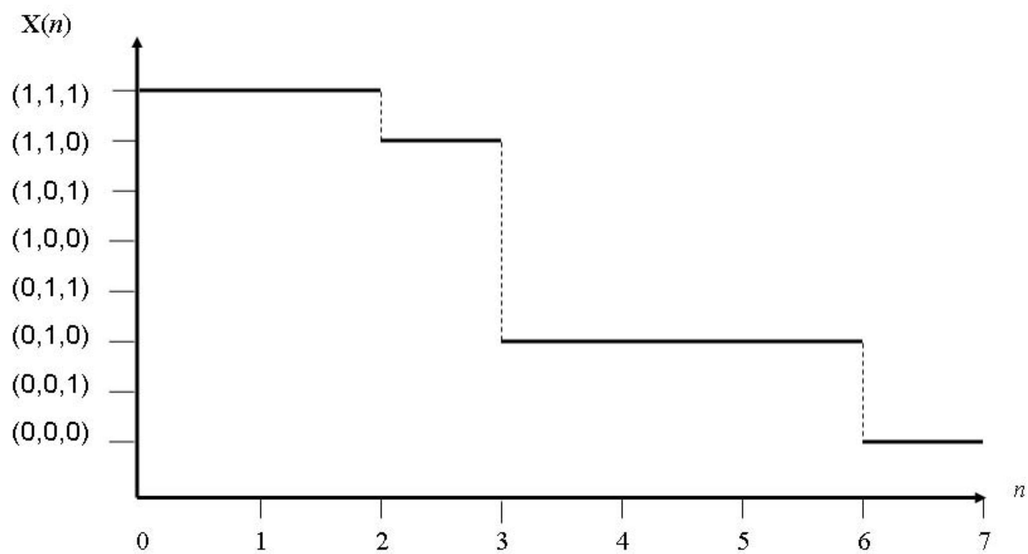

Figure 4.1 Possible sample path for $\{X(n) : n \geq 0\}$ when no maintenance is performed for example 1.

The expected lifetime for this satellite is assumed to be 5 years. It is reasonable to assume that each subsystem on the satellite has an expected lifetime at least as long as the entire satellite. Therefore, the expected lifetime of function 1 is assumed to be 5.5 years, the expected lifetime of function 2 is assumed to be 5.25 years, and

the expected lifetime of function 3 is assumed to be 6.5 years. In other words, if quarterly inspections are performed then the failure rates of each function are as given in Table 4.4.

Table 4.4 Failure rates of each function for example 1.

Failure Rate	*Value*
λ_1	0.045455
λ_2	0.047619
λ_3	0.038462

Using the failure rates of Table 4.4 and the assumption of exponential lifetimes, the transition probability matrix can be easily determined to be

$$\mathbf{P} = \begin{bmatrix} R_1R_2R_3 & R_1R_2F_3 & R_1F_2R_3 & R_1F_2F_3 & F_1R_2R_3 & F_1R_2F_3 & F_1F_2R_3 & F_1F_2F_3 \\ 0 & R_1R_2 & 0 & R_1F_2 & 0 & F_1R_2 & 0 & F_1F_2 \\ 0 & 0 & R_1R_3 & R_1F_3 & 0 & 0 & F_1R_3 & F_1F_3 \\ 0 & 0 & 0 & R_1 & 0 & 0 & 0 & F_1 \\ 0 & 0 & 0 & 0 & R_2R_3 & R_2F_3 & F_2R_3 & F_2F_3 \\ 0 & 0 & 0 & 0 & 0 & R_2 & 0 & F_2 \\ 0 & 0 & 0 & 0 & 0 & 0 & R_3 & F_3 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.2}$$

where $F_m = 1 - e^{-\lambda_m}$ and $R_m = e^{-\lambda_m}$, $m = 1, 2, 3$. Substituting the numerical values into Equation (4.2) yields

$$\mathbf{P} = \begin{bmatrix} 0.87675 & 0.03438 & 0.04276 & 0.00168 & 0.04077 & 0.00160 & 0.00199 & 0.00008 \\ 0 & 0.91113 & 0 & 0.04444 & 0 & 0.04237 & 0 & 0.00207 \\ 0 & 0 & 0.91951 & 0.03606 & 0 & 0 & 0.04276 & 0.00168 \\ 0 & 0 & 0 & 0.95556 & 0 & 0 & 0 & 0.04444 \\ 0 & 0 & 0 & 0 & 0.91752 & 0.03598 & 0.04475 & 0.00175 \\ 0 & 0 & 0 & 0 & 0 & 0.95350 & 0 & 0.04650 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.96227 & 0.03773 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{4.3}$$

The transition diagram is given in Figure 4.2. Note that transitions may only occur to the same state or a higher indexed (more degraded) state because no maintenance is being performed.

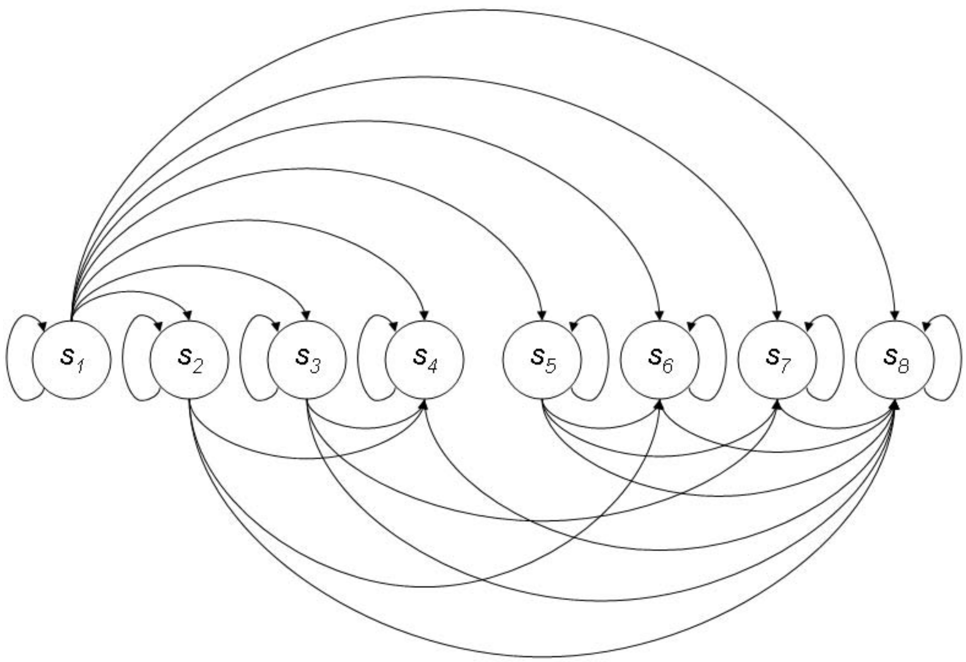

Figure 4.2 State transition diagram depicting all possible transitions, example 1.

4.2.2 One-Satellite MDP Formulation

Now that the stochastic degradation process for the one-satellite example has been described, we formulate the optimization problem as a Markov decision process which can be solved to find an optimal maintenance policy.

The inspections of the system will occur quarterly for five years ($N = 20$). The set of decision epochs is given by

$$\mathcal{N} = \{1, 2, \ldots, 20\}. \tag{4.4}$$

The state space for the MDP model is identical to the state space described in the degradation process. There are three functions, each having the ability to either be operational, $X_m(n) = 1$, or non-operational, $X_m(n) = 0$. This produces the random state vector

$$\mathbf{X}(n) = (X_1(n), X_2(n), X_3(n)).$$

The complete list of states can be referenced in Table 4.3.

Suppose the decision maker feels it is critical to have at least one function operational. In other words, when all three functions fail, some maintenance action must be performed. Furthermore, when all functions are operational, no on-orbit repairs should be performed. Using this information, an action set can be established. The feasible actions for this scenario are listed in Table 4.5.

Table 4.5 Feasible maintenance actions for example 1.

Action Set	*Action*	*Definition*
$\mathcal{A}_1$	$a_{1,1}$	do nothing
	$a_{1,3}$	replace
$\mathcal{A}_2 - \mathcal{A}_7$	$a_{s,1}$	do nothing
	$a_{s,2}$	on orbit repair
	$a_{s,3}$	replace
$\mathcal{A}_8$	$a_{8,2}$	on orbit repair
	$a_{8,3}$	replace

The transition probabilities require knowledge of the probability of successfully repairing function m, denoted by H_m, and the probability function m survives replacement, denoted by G_m. The notional repair and replacement probabilities for each function are summarized in Table 4.6.

Using the successful repair and replacement probabilities from Table 4.6 and the transition probabilities from Equation (4.3), the transition probabilities for the MDP can be computed. Choosing a maintenance action is equivalent to choosing the transition probability matrix by which the satellite will evolve during the next inter-inspection period. For this example, there are three possible actions indicating that

Table 4.6 Repair and replacement probabilities for each function for example 1.

Function	H_m (*Repair Probability*)	G_m (*Replace Probability*)
Function 1	0.95	0.975
Function 2	0.96	0.94
Function 3	0.97	0.98

the satellite will evolve stochastically according to one of three transition probability matrices at each decision epoch. If action 1 (do nothing) is chosen, then the satellite will evolve according to the following transition probability matrix for the next inter-inspection interval:

$$\mathbf{P} = \begin{bmatrix} R_1R_2R_3 & R_1R_2F_3 & R_1F_2R_3 & R_1F_2F_3 & F_1R_2R_3 & F_1R_2F_3 & F_1F_2R_3 & F_1F_2F_3 \\ 0 & R_1R_2 & 0 & R_1F_2 & 0 & F_1R_2 & 0 & F_1F_2 \\ 0 & 0 & R_1R_3 & R_1F_3 & 0 & 0 & F_1R_3 & F_1F_3 \\ 0 & 0 & 0 & R_1 & 0 & 0 & 0 & F_1 \\ 0 & 0 & 0 & 0 & R_2R_3 & R_2F_3 & F_2R_3 & F_2F_3 \\ 0 & 0 & 0 & 0 & 0 & R_2 & 0 & F_2 \\ 0 & 0 & 0 & 0 & 0 & 0 & R_3 & F_3 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \quad (4.5)$$

Substituting the numerical values into (4.5) yields

$$\mathbf{P} = \begin{bmatrix} 0.87675 & 0.03438 & 0.04276 & 0.00168 & 0.04077 & 0.00160 & 0.00199 & 0.00008 \\ 0 & 0.91113 & 0 & 0.04444 & 0 & 0.04237 & 0 & 0.00207 \\ 0 & 0 & 0.91951 & 0.03605 & 0 & 0 & 0.04276 & 0.00168 \\ 0 & 0 & 0 & 0.95556 & 0 & 0 & 0 & 0.04444 \\ 0 & 0 & 0 & 0 & 0.91752 & 0.03598 & 0.04475 & 0.00175 \\ 0 & 0 & 0 & 0 & 0 & 0.95350 & 0 & 0.04650 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.96227 & 0.03773 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \quad (4.6)$$

If action 2 (on-orbit repair) is chosen, then the satellite will evolve according to the following transition probability matrix for the next inter-inspection interval:

$$
\mathbf{P} = \begin{bmatrix}
R_1R_2R_3 & R_1R_2F_3 & R_1F_2R_3 & R_1F_2F_3 & F_1R_2R_3 & F_1R_2F_3 & F_1F_2R_3 & F_1F_2F_3 \\
R_1R_2H_3 & R_1R_2\bar{H}_3 & R_1F_2H_3 & R_1F_2\bar{H}_3 & F_1R_2H_3 & F_1R_2\bar{H}_3 & F_1F_2H_3 & F_1F_2\bar{H}_3 \\
R_1H_2R_3 & R_1H_2F_3 & R_1\bar{H}_2R_3 & R_1\bar{H}_2F_3 & F_1H_2R_3 & F_1\bar{H}_2F_3 & F_1\bar{H}_2R_3 & F_1\bar{H}_2F_3 \\
R_1H_2H_3 & R_1H_2\bar{H}_3 & R_1H_2\bar{H}_3 & R_1\bar{H}_2\bar{H}_3 & F_1H_2H_3 & F_1H_2\bar{H}_3 & F_1\bar{H}_2H_3 & F_1\bar{H}_2\bar{H}_3 \\
H_1R_2R_3 & H_1R_2F_3 & \bar{H}_1F_2R_3 & H_1F_2F_3 & \bar{H}_1R_2R_3 & \bar{H}_1R_2F_3 & \bar{H}_1F_2R_3 & \bar{H}_1F_2F_3 \\
H_1R_2H_3 & H_1R_2\bar{H}_3 & H_1F_2H_3 & H_1F_2\bar{H}_3 & \bar{H}_1R_2H_3 & \bar{H}_1R_2\bar{H}_3 & \bar{H}_1F_2H_3 & \bar{H}_1F_2\bar{H}_3 \\
H_1H_2R_3 & H_1H_2F_3 & H_1\bar{H}_2R_3 & H_1\bar{H}_2F_3 & \bar{H}_1H_2R_3 & \bar{H}_1H_2F_3 & \bar{H}_1\bar{H}_2R_3 & \bar{H}_1\bar{H}_2F_3 \\
H_1H_2H_3 & H_1H_2\bar{H}_3 & H_1\bar{H}_2H_3 & H_1\bar{H}_2\bar{H}_3 & \bar{H}_1H_2H_3 & \bar{H}_1H_2\bar{H}_3 & \bar{H}_1\bar{H}_2H_3 & \bar{H}_1\bar{H}_2\bar{H}_3
\end{bmatrix}. \tag{4.7}
$$

Notice that the first row of this transition probability matrix is equivalent to that of Equation (4.5). This is because of the assumption that only non-operational functions will be repaired. If all functions are operational, then no functions will be repaired and the system will evolve according to the underlying stochastic degradation process. Substituting the numerical values into (4.7) yields

$$
\mathbf{P} = \begin{bmatrix}
0.87675 & 0.03438 & 0.04276 & 0.00168 & 0.04077 & 0.00160 & 0.00199 & 0.00008 \\
0.88379 & 0.02733 & 0.04310 & 0.00133 & 0.04110 & 0.00127 & 0.00200 & 0.00006 \\
0.88273 & 0.03461 & 0.03678 & 0.00144 & 0.04105 & 0.00161 & 0.00171 & 0.00007 \\
0.88982 & 0.02752 & 0.03708 & 0.00115 & 0.04138 & 0.00128 & 0.00172 & 0.00005 \\
0.87164 & 0.03418 & 0.04251 & 0.00167 & 0.04588 & 0.00180 & 0.00224 & 0.00009 \\
0.87865 & 0.02717 & 0.04285 & 0.00133 & 0.04624 & 0.00143 & 0.00226 & 0.00007 \\
0.87759 & 0.03441 & 0.03657 & 0.00143 & 0.04619 & 0.00181 & 0.00192 & 0.00008 \\
0.88464 & 0.02736 & 0.03686 & 0.00114 & 0.04656 & 0.00144 & 0.00194 & 0.00006
\end{bmatrix}. \tag{4.8}
$$

If action 3 (replace) is chosen, then the satellite will evolve according to the following transition probability matrix for the next inter-inspection interval:

$$\mathbf{P} = \begin{bmatrix} G_1G_2G_3 & G_1G_2\bar{G}_3 & G_1\bar{G}_2G_3 & G_1\bar{G}_2\bar{G}_3 & \bar{G}_1G_2G_3 & \bar{G}_1G_2\bar{G}_3 & \bar{G}_1\bar{G}_2G_3 & \bar{G}_1\bar{G}_2\bar{G}_3 \\ G_1G_2G_3 & G_1G_2\bar{G}_3 & G_1\bar{G}_2G_3 & G_1\bar{G}_2\bar{G}_3 & \bar{G}_1G_2G_3 & \bar{G}_1G_2\bar{G}_3 & \bar{G}_1\bar{G}_2G_3 & \bar{G}_1\bar{G}_2\bar{G}_3 \\ G_1G_2G_3 & G_1G_2\bar{G}_3 & G_1\bar{G}_2G_3 & G_1\bar{G}_2\bar{G}_3 & \bar{G}_1G_2G_3 & \bar{G}_1G_2\bar{G}_3 & \bar{G}_1\bar{G}_2G_3 & \bar{G}_1\bar{G}_2\bar{G}_3 \\ G_1G_2G_3 & G_1G_2\bar{G}_3 & G_1\bar{G}_2G_3 & G_1\bar{G}_2\bar{G}_3 & \bar{G}_1G_2G_3 & \bar{G}_1G_2\bar{G}_3 & \bar{G}_1\bar{G}_2G_3 & \bar{G}_1\bar{G}_2\bar{G}_3 \\ G_1G_2G_3 & G_1G_2\bar{G}_3 & G_1\bar{G}_2G_3 & G_1\bar{G}_2\bar{G}_3 & \bar{G}_1G_2G_3 & \bar{G}_1G_2\bar{G}_3 & \bar{G}_1\bar{G}_2G_3 & \bar{G}_1\bar{G}_2\bar{G}_3 \\ G_1G_2G_3 & G_1G_2\bar{G}_3 & G_1\bar{G}_2G_3 & G_1\bar{G}_2\bar{G}_3 & \bar{G}_1G_2G_3 & \bar{G}_1G_2\bar{G}_3 & \bar{G}_1\bar{G}_2G_3 & \bar{G}_1\bar{G}_2\bar{G}_3 \\ G_1G_2G_3 & G_1G_2\bar{G}_3 & G_1\bar{G}_2G_3 & G_1\bar{G}_2\bar{G}_3 & \bar{G}_1G_2G_3 & \bar{G}_1G_2\bar{G}_3 & \bar{G}_1\bar{G}_2G_3 & \bar{G}_1\bar{G}_2\bar{G}_3 \\ G_1G_2G_3 & G_1G_2\bar{G}_3 & G_1\bar{G}_2G_3 & G_1\bar{G}_2\bar{G}_3 & \bar{G}_1G_2G_3 & \bar{G}_1G_2\bar{G}_3 & \bar{G}_1\bar{G}_2G_3 & \bar{G}_1\bar{G}_2\bar{G}_3 \end{bmatrix}. \tag{4.9}$$

Substituting the numerical values into (4.9) yields

$$\mathbf{P} = \begin{bmatrix} 0.89817 & 0.01833 & 0.05733 & 0.00117 & 0.02303 & 0.00047 & 0.00147 & 0.00003 \\ 0.89817 & 0.01833 & 0.05733 & 0.00117 & 0.02303 & 0.00047 & 0.00147 & 0.00003 \\ 0.89817 & 0.01833 & 0.05733 & 0.00117 & 0.02303 & 0.00047 & 0.00147 & 0.00003 \\ 0.89817 & 0.01833 & 0.05733 & 0.00117 & 0.02303 & 0.00047 & 0.00147 & 0.00003 \\ 0.89817 & 0.01833 & 0.05733 & 0.00117 & 0.02303 & 0.00047 & 0.00147 & 0.00003 \\ 0.89817 & 0.01833 & 0.05733 & 0.00117 & 0.02303 & 0.00047 & 0.00147 & 0.00003 \\ 0.89817 & 0.01833 & 0.05733 & 0.00117 & 0.02303 & 0.00047 & 0.00147 & 0.00003 \\ 0.89817 & 0.01833 & 0.05733 & 0.00117 & 0.02303 & 0.00047 & 0.00147 & 0.00003 \end{bmatrix}. \tag{4.10}$$

The notional cost to perform on-orbit repairs is assumed to be \$450 million for the space vehicle plus the function-specific repair costs. The specific repair cost of function 1 is \$35 million, the specific repair cost of function 2 is \$15 million, and the specific repair cost of function 3 is \$20 million. Note that economies of scale exist for on orbit repairs, that is, it is more costly to repair each function at different times than it is to repair multiple functions at the same time. The cost of a satellite replacement (unit cost) is assumed to be \$500 million. The notional penalty costs, assumed to be in dollars, are given in Table 4.7.

Table 4.7 Penalty costs, example 1.

State	*Penalty Cost* $C_p(s)$
s_1	\$0
s_2	\$200 million
s_3	\$500 million
s_4	\$600 million
s_5	\$300 million
s_6	\$400 million
s_7	\$400 million
s_8	\$700 million

The expected rewards are obtained using Equation (3.28). In the case of $K = 1$, $|\mathcal{S}| = 8$ this equation simplifies to

$$r_n(s, a) = r(a) + \sum_{j=1}^{8} C_p(s_j) p_n(j|s, a). \tag{4.11}$$

The rewards for the first $N - 1$ decision epochs are listed in Table 4.8.

Because these rewards look into the next time interval to compute the expected rewards, and because there are no decisions made at the final decision epoch, $N = 20$, there is no terminal reward so that $r_{20}(s_1) = r_{20}(s_2) = \cdots = r_{20}(s_8) = 0$.

4.2.3 One-Satellite Optimality Results

The one-satellite example was solved by means of the backward induction algorithm coded in MATLAB®. The optimal policy obtained using this method is presented in Table 4.9. The policy given in Table 4.9 lists the optimal decision rule for each decision epoch (Note that there is no decision made at the final decision epoch, $N = 20$). Each decision rule gives the optimal action to take for every possible state the system may be found in at that decision epoch. For example, at 2 years the optimal decision rule is the column of numbers below $n = 8$, $(1, 2, 2, 2, 3, 3, 3, 3)^T$. This decision rule is interpreted to mean that at time $n = 8$ the action that produces the minimum expected total loss over the final three years is:

Table 4.8 Expected rewards given the state and action taken for the first $N-1$ decision epochs as computed by Equation (3.28), example 1.

Action	*State*	*Reward*	*Value (millions)*
Do Nothing	s_1	$r_n(s_1, a_{1,1})$	\$42.98255
Do Nothing	s_2	$r_n(s_2, a_{2,1})$	\$227.28197
Do Nothing	s_3	$r_n(s_3, a_{3,1})$	\$499.66476
Do Nothing	s_4	$r_n(s_4, a_{4,1})$	\$604.44370
Do Nothing	s_5	$r_n(s_5, a_{5,1})$	\$308.77436
Do Nothing	s_6	$r_n(s_6, a_{6,1})$	\$413.95091
Do Nothing	s_7	$r_n(s_7, a_{7,1})$	\$411.31939
Do Nothing	s_8	$r_n(s_8, a_{8,1})$	N/A
Repair	s_1	$r_n(s_1, a_{1,2})$	N/A
Repair	s_2	$r_n(s_2, a_{2,2})$	\$511.50181
Repair	s_3	$r_n(s_3, a_{3,2})$	\$504.86788
Repair	s_4	$r_n(s_4, a_{4,2})$	\$523.38278
Repair	s_5	$r_n(s_5, a_{5,2})$	\$529.52992
Repair	s_6	$r_n(s_6, a_{6,2})$	\$548.05288
Repair	s_7	$r_n(s_7, a_{7,2})$	\$541.42931
Repair	s_8	$r_n(s_8, a_{8,2})$	\$559.94800
Replace	All	$r_n(s_., a_{.,3})$	\$540.73900

- action 1 (do nothing) if the system is found to be in state s_1;
- action 2 (on-orbit repair) if the system is found to be in states s_2, s_3, or s_4;
- action 3 (replacement) if the system is found to be in states s_5, s_6, s_7, or s_8.

The total expected loss that results from the optimal policy depends on the initial state of the satellite. Table 4.10 gives the minimal loss for each possible initial state. For example, if the system is initially observed to be in state s_3 then the expected loss over the 5 year lifetime of the constellation is \$2298.20 million.

4.2.4 One-Satellite Sensitivity Analysis

The optimal policy and resulting optimal value are the result of a set of fixed input parameters. These parameters are assumed to be accurate, but in reality this may not be the case. For this reason it is important to know how sensitive the

Table 4.9 Optimal maintenance policy for example 1.

State	Decision Epoch (n)																		
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
s_1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
s_2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	1	1
s_3	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	1
s_4	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
s_5	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	2	1
s_6	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	1
s_7	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	1
s_8	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3

Table 4.10 Minimum expected total loss for every possible initial state (*millions*) for example 1.

State	*Minimum Loss*	*State*	*Minimum Loss*
s_1	$1839.06	s_5	$2326.33
s_2	$2304.30	s_6	$2326.33
s_3	$2298.20	s_7	$2326.33
s_4	$2313.41	s_8	$2326.33

optimal value *and* optimal policy are to changes in the input parameters. In some instances, it may be more beneficial to implement a suboptimal policy if it is less sensitive to changes in the parameters. For this analysis, we will assume that the system is found to be in the fully capable state initially, that is $\mathbf{X}(1) = (1, 1, 1)$.

Of particular interest are the conditions for which the optimal policy includes both on-orbit repairs and satellite replacements, henceforth referred to as *mixed policies*. As mentioned in the MDP formulation, an on-orbit repair cost consists of the space vehicle cost plus the function specific repair costs. For this analysis, the function specific repair costs are assumed to be fixed for each function. However, the space vehicle cost is varied from $10 million to $1 billion while the replacement cost is held constant to see its effect on the minimum total expected loss. Likewise, the unit cost for a satellite replacement cost is varied from $10 million to $1 billion

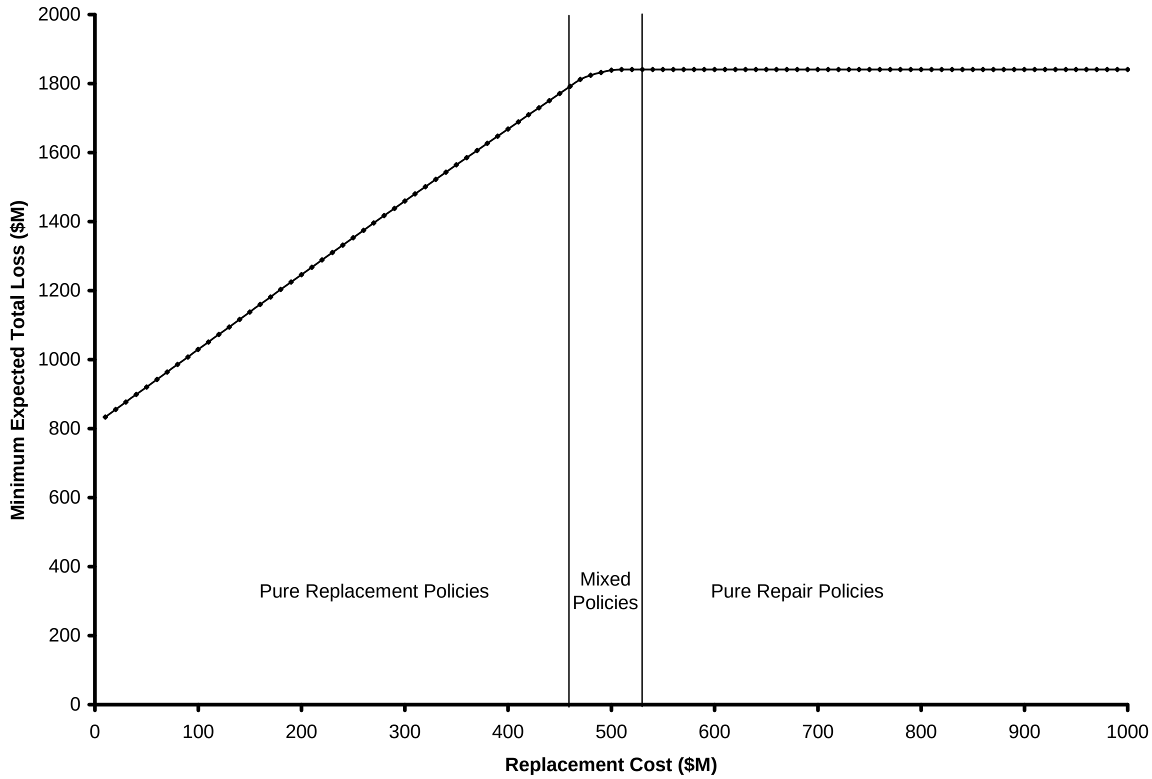

Figure 4.3 One-way sensitivity analysis of the minimum total expected loss when varying the replacement cost from $10 million to $1 billion while holding the space vehicle cost at $450 million, example 1.

while the space vehicle cost is held constant to see its effect on the minimum total expected loss.

Figure 4.3 depict graphically the change in the minimum total expected loss caused by varying the unit cost of a satellite with all else held constant. Figure 4.4 depicts graphically the change in the minimum total expected loss caused by varying the cost of the space vehicle with all else held constant. These two figures illustrate that mixed policies are optimal only when the difference between on-orbit repair costs and satellite replacement costs is small. If the on-orbit repair cost is significantly greater than the replacement cost, the optimal policy consists only of replacement actions. Likewise, if the replacement cost is significantly greater than the on-orbit repair cost, the optimal policy consists only of on-orbit repair actions. Figure 4.4 illustrates that when the replacement cost is fixed at $500 million, mixed policies

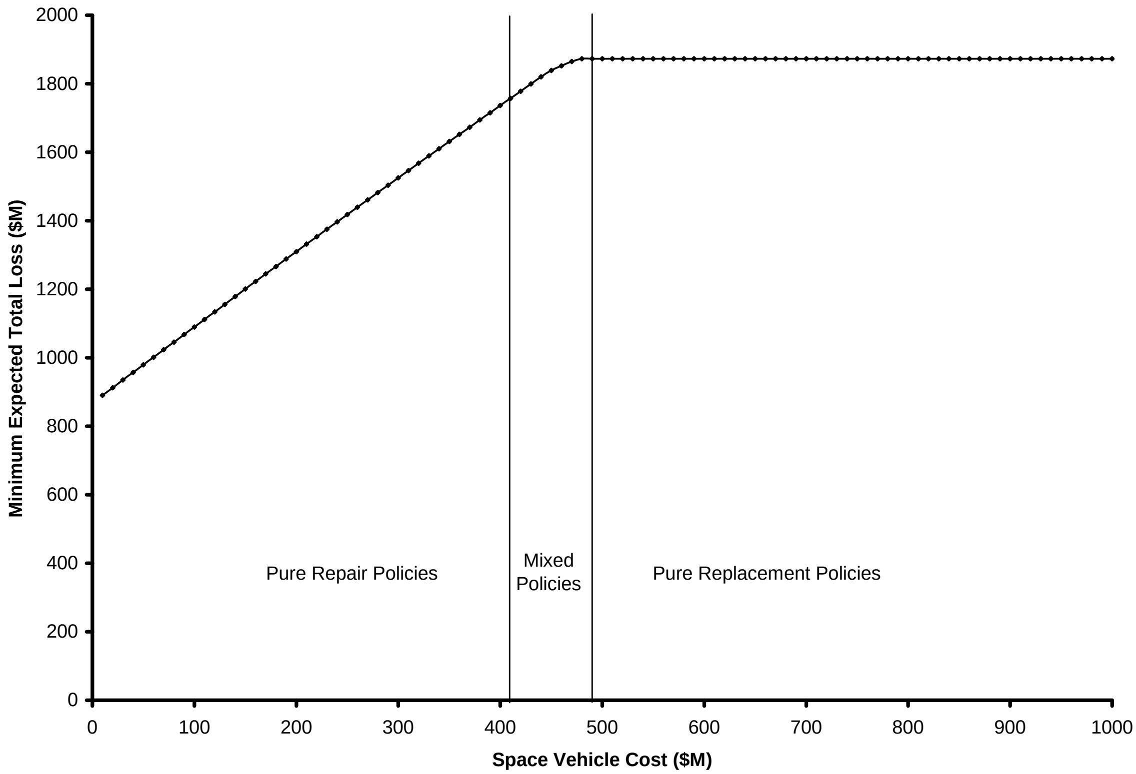

Figure 4.4 One-way sensitivity analysis of the minimum total expected loss when varying the space vehicle cost from \$10 million to \$1 billion while holding the replacement cost at \$500 million, example 1.

are optimal when the space vehicle cost is approximately between \$400 million and \$500 million, or 80% to 100% of the replacement cost.

To gain further insight, another one-way analysis was done to examine the space vehicle cost to replacement cost ratios for which mixed policies are optimal when the replacement cost is fixed at \$500 million. As can be seen in Figure 4.5, mixed policies are optimal when this ratio is between 0.84 and 0.98. It is worth mentioning that under the conditions of this analysis, on-orbit repairs appear in the optimal policy for all ratios less than 0.98.

Next, we consider a two-way sensitivity analysis to determine the maintenance cost ratios for which mixed policies are optimal. To do this, the replacement cost is

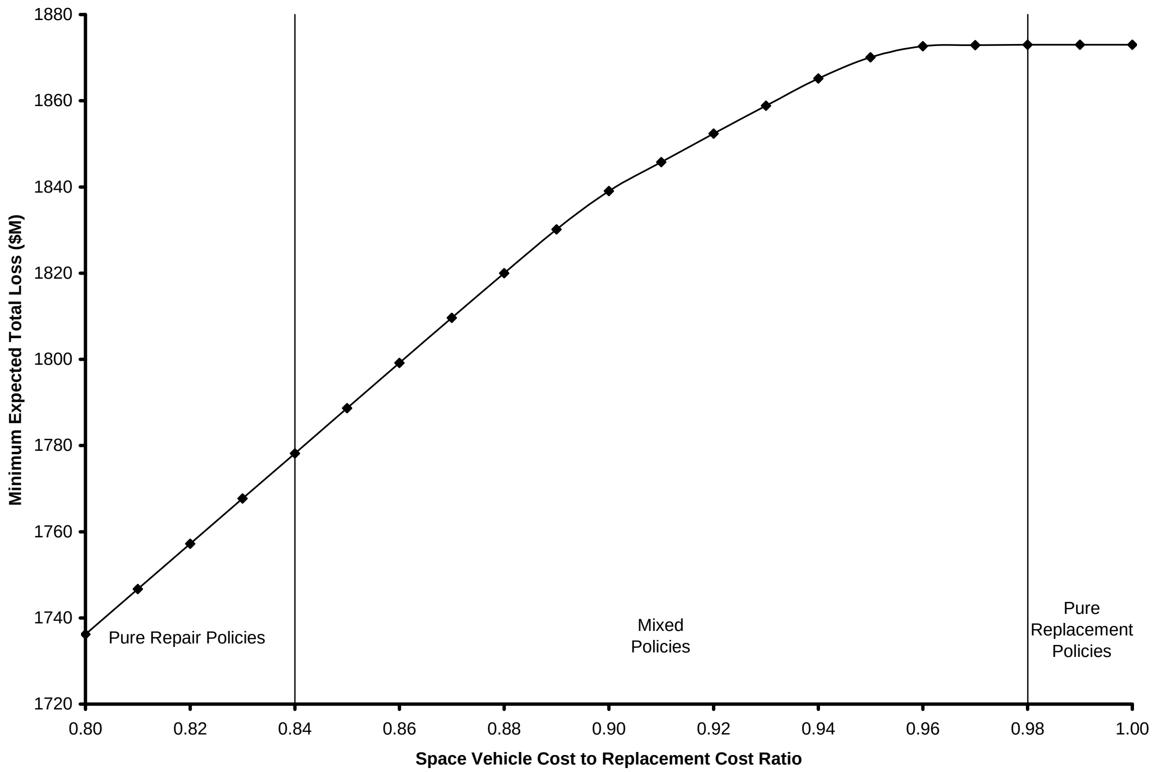

Figure 4.5 One-way sensitivity analysis of the minimum total expected loss when varying the space vehicle cost to replacement cost ratio from 0.80 to 1.0 while holding the replacement cost at $500 million, example 1.

varied from $50 million to $1 billion in $50 million increments while the space vehicle cost is varied from 0 to 100% of the replacement cost.

As can be seen in Figure 4.6, the cost ratios for which mixed policies are optimal depends on the magnitude of the maintenance costs. The range of cost ratios for which mixed policies are optimal is wider for small magnitudes than for large magnitudes. Of notable interest is the fact that when the replacement cost of the satellite is $50 million, mixed policies are optimal for all cost ratios less than 0.72. We performed further analysis to determine that when the replacement cost is less than or equal to $72 million, there is a cost ratio r such that all policies for cost ratios greater than r are pure replacement policies and all policies for cost ratios less than or equal to r are mixed policies. This implies that under the conditions of this

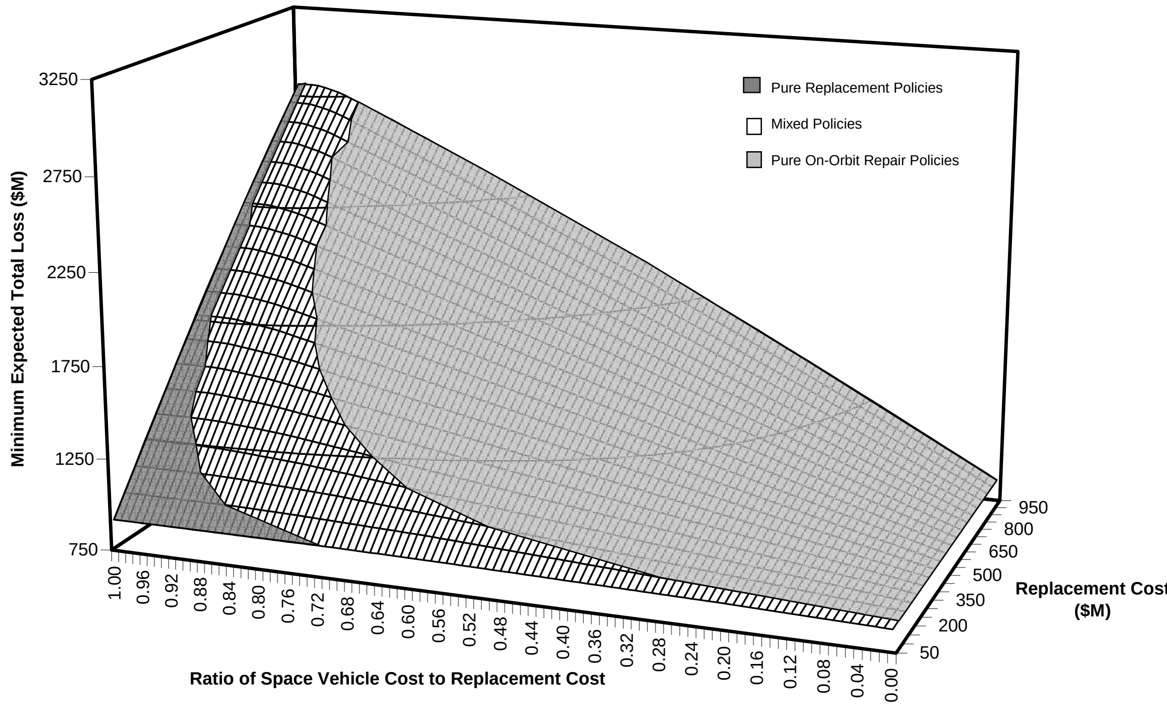

Figure 4.6 Two-way sensitivity analysis of the minimum total expected loss when varying the replacement cost from \$50 million to \$1 billion while varying the ratio of the space vehicle cost to replacement cost from 0.0 to 1.0, example 1.

example, satellite replacements will be included in the optimal policy whenever the cost of replacement is less than or equal to \$72 million.

4.3 Three-Satellite Example

In the second example, consider a satellite constellation consisting of three satellites. Satellites 1 and 2 each have two functions and satellite 3 has three functions. Also, suppose that a decision maker desires an optimal maintenance policy for this satellite constellation over the next 5 years. The constellation will be inspected every quarter, and a decision will be made to either do nothing, repair, or replace each satellite. The constellation contains three satellites ($K = 3$) with two functions for each of the first two satellites ($M^{(1)} = 2$, $M^{(2)} = 2$) and three functions on the third satellite ($M^{(3)} = 3$); therefore, there are $2^{2+2+3} = 128$ possible degradation

levels ($|\mathcal{S}| = 128$) for the constellation. Since four inspections will be done every year for five years there are a total of 20 decision epochs ($N = 20$).

4.3.1 Three-Satellite Degradation Process

The degradation process can be viewed as three separate degradation processes because of the assumption of mutual independence among satellite degradation processes. Each individual satellite has its own state space and degradation process. Two of the satellites in this example have two functions and thus have four possible degradation states. The remaining satellite has three functions giving it eight possible degradation states. Table 4.11 illustrates each satellite's individual state space.

Table 4.11 Individual satellite state spaces, example 2.

Satellite 1	*Satellite 2*	*Satellite 3*
$\mathcal{S}^{(1)} = \left\{ \begin{array}{lcl} s_1^{(1)} & = & (1,1) \\ s_2^{(1)} & = & (1,0) \\ s_3^{(1)} & = & (0,1) \\ s_4^{(1)} & = & (0,0) \end{array} \right\}$	$\mathcal{S}^{(2)} = \left\{ \begin{array}{lcl} s_1^{(2)} & = & (1,1) \\ s_2^{(2)} & = & (1,0) \\ s_3^{(2)} & = & (0,1) \\ s_4^{(2)} & = & (0,0) \end{array} \right\}$	$\mathcal{S}^{(3)} = \left\{ \begin{array}{lcl} s_1^{(3)} & = & (1,1,1) \\ s_2^{(3)} & = & (1,1,0) \\ s_3^{(3)} & = & (1,0,1) \\ s_4^{(3)} & = & (1,0,0) \\ s_5^{(3)} & = & (0,1,1) \\ s_6^{(3)} & = & (0,1,0) \\ s_7^{(3)} & = & (0,0,1) \\ s_8^{(3)} & = & (0,0,0) \end{array} \right\}$

The overall state space of the three satellite constellation model is a three component row vector of each individual satellite's state given by

$$\mathbf{X}(n) = (\mathbf{X}^{(1)}(n), \mathbf{X}^{(2)}(n), \mathbf{X}^{(3)}(n)) \tag{4.12}$$

where $\mathbf{X}^{(k)}(n) \in \mathcal{S}^{(k)}, k = 1, 2, 3$. Equation (4.12) can also be written as

$$\mathbf{X}(n) = \left((\mathbf{X}_1^{(1)}(n), \mathbf{X}_2^{(1)}(n)),\ (\mathbf{X}_1^{(2)}(n), \mathbf{X}_2^{(2)}(n)),\ (\mathbf{X}_1^{(3)}(n), \mathbf{X}_2^{(3)}(n), \mathbf{X}_3^{(3)}(n)) \right). \tag{4.13}$$

Table 4.12 gives the enumerated state space for the satellite constellation. The 'state' columns give the label for the state, s_i where i is the index assigned to that state. The 'vector' columns give the associated state vector where each component is the index of that individual satellite's state. For example, $(1, 1, 1)$ means that state s_1 of the constellation represents the vector $(s_1^{(1)}, s_1^{(2)}, s_1^{(3)})$.

Table 4.12 Enumerated state space, example 2.

State	*Vector*	*State*	*Vector*	*State*	*Vector*	*State*	*Vector*
s_1	$(1,1,1)$	s_{33}	$(2,1,1)$	s_{65}	$(3,1,1)$	s_{97}	$(4,1,1)$
s_2	$(1,1,2)$	s_{34}	$(2,1,2)$	s_{66}	$(3,1,2)$	s_{98}	$(4,1,2)$
s_3	$(1,1,3)$	s_{35}	$(2,1,3)$	s_{67}	$(3,1,3)$	s_{99}	$(4,1,3)$
s_4	$(1,1,4)$	s_{36}	$(2,1,4)$	s_{68}	$(3,1,4)$	s_{100}	$(4,1,4)$
s_5	$(1,1,5)$	s_{37}	$(2,1,5)$	s_{69}	$(3,1,5)$	s_{101}	$(4,1,5)$
s_6	$(1,1,6)$	s_{38}	$(2,1,6)$	s_{70}	$(3,1,6)$	s_{102}	$(4,1,6)$
s_7	$(1,1,7)$	s_{39}	$(2,1,7)$	s_{71}	$(3,1,7)$	s_{103}	$(4,1,7)$
s_8	$(1,1,8)$	s_{40}	$(2,1,8)$	s_{72}	$(3,1,8)$	s_{104}	$(4,1,8)$
s_9	$(1,2,1)$	s_{41}	$(2,2,1)$	s_{73}	$(3,2,1)$	s_{105}	$(4,2,1)$
s_{10}	$(1,2,2)$	s_{42}	$(2,2,2)$	s_{74}	$(3,2,2)$	s_{106}	$(4,2,2)$
s_{11}	$(1,2,3)$	s_{43}	$(2,2,3)$	s_{75}	$(3,2,3)$	s_{107}	$(4,2,3)$
s_{12}	$(1,2,4)$	s_{44}	$(2,2,4)$	s_{76}	$(3,2,4)$	s_{108}	$(4,2,4)$
s_{13}	$(1,2,5)$	s_{45}	$(2,2,5)$	s_{77}	$(3,2,5)$	s_{109}	$(4,2,5)$
s_{14}	$(1,2,6)$	s_{46}	$(2,2,6)$	s_{78}	$(3,2,6)$	s_{110}	$(4,2,6)$
s_{15}	$(1,2,7)$	s_{47}	$(2,2,7)$	s_{79}	$(3,2,7)$	s_{111}	$(4,2,7)$
s_{16}	$(1,2,8)$	s_{48}	$(2,2,8)$	s_{80}	$(3,2,8)$	s_{112}	$(4,2,8)$
s_{17}	$(1,3,1)$	s_{49}	$(2,3,1)$	s_{81}	$(3,3,1)$	s_{113}	$(4,3,1)$
s_{18}	$(1,3,2)$	s_{50}	$(2,3,2)$	s_{82}	$(3,3,2)$	s_{114}	$(4,3,2)$
s_{19}	$(1,3,3)$	s_{51}	$(2,3,3)$	s_{83}	$(3,3,3)$	s_{115}	$(4,3,3)$
s_{20}	$(1,3,4)$	s_{52}	$(2,3,4)$	s_{84}	$(3,3,4)$	s_{116}	$(4,3,4)$
s_{21}	$(1,3,5)$	s_{53}	$(2,3,5)$	s_{85}	$(3,3,5)$	s_{117}	$(4,3,5)$
s_{22}	$(1,3,6)$	s_{54}	$(2,3,6)$	s_{86}	$(3,3,6)$	s_{118}	$(4,3,6)$
s_{23}	$(1,3,7)$	s_{55}	$(2,3,7)$	s_{87}	$(3,3,7)$	s_{119}	$(4,3,7)$
s_{24}	$(1,3,8)$	s_{56}	$(2,3,8)$	s_{88}	$(3,3,8)$	s_{120}	$(4,3,8)$
s_{25}	$(1,4,1)$	s_{57}	$(2,4,1)$	s_{89}	$(3,4,1)$	s_{121}	$(4,4,1)$
s_{26}	$(1,4,2)$	s_{58}	$(2,4,2)$	s_{90}	$(3,4,2)$	s_{122}	$(4,4,2)$
s_{27}	$(1,4,3)$	s_{59}	$(2,4,3)$	s_{91}	$(3,4,3)$	s_{123}	$(4,4,3)$
s_{28}	$(1,4,4)$	s_{60}	$(2,4,4)$	s_{92}	$(3,4,4)$	s_{124}	$(4,4,4)$
s_{29}	$(1,4,5)$	s_{61}	$(2,4,5)$	s_{93}	$(3,4,5)$	s_{125}	$(4,4,5)$
s_{30}	$(1,4,6)$	s_{62}	$(2,4,6)$	s_{94}	$(3,4,6)$	s_{126}	$(4,4,6)$
s_{31}	$(1,4,7)$	s_{63}	$(2,4,7)$	s_{95}	$(3,4,7)$	s_{127}	$(4,4,7)$
s_{32}	$(1,4,8)$	s_{64}	$(2,4,8)$	s_{96}	$(3,4,8)$	s_{128}	$(4,4,8)$

The notional expected lifetime for each satellite is 5 years. It is acceptable to assume that each subsystem on each satellite has an expected lifetime at least as long as the satellite design life. In this example, functions 1 and 2 are assumed to be identical on all satellites. The expected lifetime of function 1 will be assumed to be 5.5 years, the expected lifetime of function 2 will be assumed to be 5.25 years, and the expected lifetime of function 3 will be assumed to be 6.5 years. In other words, if quarterly inspections are performed, then the failure rates of each function are as given in Table 4.13.

Table 4.13 Failure rates of each function, example 2.

Failure Rate	*Value*
λ_1	0.045455
λ_2	0.047619
λ_3	0.038462

Using the failure rates from Table 4.13 and the assumption of exponential lifetimes, the transition probability matrices for each satellite can be easily determined. The transition probability matrix for satellites 1 and 2 is given by

$$\mathbf{P} = \begin{bmatrix} R_1R_2 & R_1F_2 & F_1R_2 & F_1F_2 \\ 0 & R_1 & 0 & F_1 \\ 0 & 0 & R_2 & F_2 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{4.14}$$

and the transition probability matrix for satellite 3 is given by

$$\mathbf{P} = \begin{bmatrix} R_1R_2R_3 & R_1R_2F_3 & R_1F_2R_3 & R_1F_2F_3 & F_1R_2R_3 & F_1R_2F_3 & F_1F_2R_3 & F_1F_2F_3 \\ 0 & R_1R_2 & 0 & R_1F_2 & 0 & F_1R_2 & 0 & F_1F_2 \\ 0 & 0 & R_1R_3 & R_1F_3 & 0 & 0 & F_1R_3 & F_1F_3 \\ 0 & 0 & 0 & R_1 & 0 & 0 & 0 & F_1 \\ 0 & 0 & 0 & 0 & R_2R_3 & R_2F_3 & F_2R_3 & F_2F_3 \\ 0 & 0 & 0 & 0 & 0 & R_2 & 0 & F_2 \\ 0 & 0 & 0 & 0 & 0 & 0 & R_3 & F_3 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.15}$$

where $F_m = 1 - e^{-\lambda_m}$ and $R_m = e^{-\lambda_m}$, $m = 1, 2, 3$. Substituting the rate parameters from Table 4.13 into (4.14) and (4.15) yields

$$\mathbf{P} = \begin{bmatrix} 0.91113 & 0.04444 & 0.04237 & 0.00207 \\ 0 & 0.95556 & 0 & 0.04444 \\ 0 & 0 & 0.95350 & 0.04650 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{4.16}$$

for satellites 1 and 2, and

$$\mathbf{P} = \begin{bmatrix} 0.87675 & 0.03438 & 0.04276 & 0.00168 & 0.04077 & 0.00160 & 0.00199 & 0.00008 \\ 0 & 0.91113 & 0 & 0.04444 & 0 & 0.04237 & 0 & 0.00207 \\ 0 & 0 & 0.91951 & 0.03606 & 0 & 0 & 0.04276 & 0.00168 \\ 0 & 0 & 0 & 0.95556 & 0 & 0 & 0 & 0.04444 \\ 0 & 0 & 0 & 0 & 0.91752 & 0.03598 & 0.04475 & 0.00175 \\ 0 & 0 & 0 & 0 & 0 & 0.95350 & 0 & 0.04650 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.96227 & 0.03773 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.17}$$

for satellite 3.

The overall constellation transition probability matrix is too large to display here. It contains $2^8 = 128$ rows and columns. In other words, there are a total of $128 \times 128 = 16,384$ constellation transition probabilities. Because the satellites are assumed to be independent, the transition probabilities for the entire constellation are the product of the individual satellite transitions. For example, the probability that the constellation transitions from state $s_1 = (s_1^{(1)}, s_1^{(2)}, s_1^{(3)})$ to state $s_{33} = (s_2^{(1)}, s_1^{(2)}, s_1^{(3)})$ is given by

$$p\{s_{33}|s_1\} = p\{s_2^{(1)}|s_1^{(1)}\} \times p\{s_1^{(2)}|s_1^{(2)}\} \times p\{s_1^{(3)})|s_1^{(3)}\}. \tag{4.18}$$

Using the individual satellite transition probability matrices in Equations (4.16) and (4.17), the value of Equation (4.18) is found to be

$$\begin{aligned} p\{s_{33}|s_1\} &= R_1F_2 \times R_1R_2 \times R_1R_2R_3 \\ &= 0.04444 \times 0.91113 \times 0.87675 \\ &= 0.03550. \end{aligned}$$

Some sample constellation transition probabilities are given in Table 4.14.

Table 4.14 Sample constellation degradation transition probabilities, example 2.

Probability	*Transition*	*Formula*	*Value*
$p\{s_2\|s_1\}$	$(s_1^{(1)}, s_1^{(2)}, s_1^{(3)}) \to (s_1^{(1)}, s_1^{(2)}, s_2^{(3)})$	$R_1R_2 \times R_1R_2 \times R_1R_2F_3$	0.02854
$p\{s_{47}\|s_1\}$	$(s_1^{(1)}, s_1^{(2)}, s_1^{(3)}) \to (s_2^{(1)}, s_2^{(2)}, s_7^{(3)})$	$R_1F_2 \times R_1F_2 \times F_1F_2R_3$	3.92650e-6
$p\{s_{47}\|s_{47}\}$	$(s_2^{(1)}, s_2^{(2)}, s_7^{(3)}) \to (s_2^{(1)}, s_2^{(2)}, s_7^{(3)})$	$R_1 \times R_1 \times R_3$	0.87865
$p\{s_{47}\|s_{111}\}$	$(s_2^{(1)}, s_2^{(2)}, s_7^{(3)}) \to (s_4^{(1)}, s_2^{(2)}, s_7^{(3)})$	$F_1 \times R_1 \times R_3$	0.04086

4.3.2 Three-Satellite MDP Formulation

The inspections of the system will occur quarterly for five years. In other words $N = 20$. The set of decision epochs is given by

$$\mathcal{N} = \{1, 2, \ldots, 20\}. \tag{4.19}$$

The state space for the MDP model is identical to the state space described in the degradation process. There are three satellites, two satellites have two functions, and one satellite has three functions. Each function has the ability to either be operational, $X_m^{(k)}(n) = 1$, or non-operational, $X_m^{(k)}(n) = 0$. This produces the state vector

$$\mathbf{X}(n) = (\mathbf{X}^{(1)}(n), \mathbf{X}^{(2)}(n), \mathbf{X}^{(3)}(n)) \tag{4.20}$$

where $\mathbf{X}^{(k)}(n) \in \mathcal{S}^{(k)}, k = 1, 2, 3$. Equation (4.20) can also be written as

$$\mathbf{X}(n) = \Big((\mathbf{X}_1^{(1)}(n), \mathbf{X}_2^{(1)}(n)),\ (\mathbf{X}_1^{(2)}(n), \mathbf{X}_2^{(2)}(n)),\ (\mathbf{X}_1^{(3)}(n), \mathbf{X}_2^{(3)}(n), \mathbf{X}_3^{(3)}(n))\Big)\,. \tag{4.21}$$

The enumerated state space can be referenced in Table 4.12.

There are three satellites, $K = 3$, and there are three possible actions for each satellite; do nothing, on orbit repair, or replace. Therefore, there are $3^3 = 27$ total combinations of actions at any decision epoch unless some actions are deemed infeasible by the decision maker. As in the single satellite example, when all functions of a satellite are operational no on-orbit repairs may be performed on that satellite. Also, the decision maker eliminates the possibility of doing nothing when all functions of all satellites are non-operational. The enumerated action space is too large to display here. There are 128 states each having a subset of 27 associated actions. The upper bound for the total number of actions in the action set is $128 \times 27 = 3,456$. As an example, the action space for state s_{11} is given in Table 4.15. Recall that s_{11} means that satellite 1 is in state $s_1^{(1)} = (1, 1)$, satellite 2 is in state $s_2^{(2)} = (1, 0)$, and satellite 3 is in state $s_3^{(3)} = (1, 0, 1)$; therefore, on-orbit repairs are not allowed for satellite 1.

The transition probabilities require knowledge of the probability that function m is successfully repaired (H_m), and the probability that function m survives replacement (G_m). The notional repair and replacement probabilities for each function are summarized in Table 4.16.

Using the successful repair and replacement probabilities from Table 4.16 and the transition probabilities of Equations (4.16) and (4.17), the transition probabilities for the MDP can be computed. Choosing a maintenance action is equivalent to choosing the transition probability matrix by which the constellation will evolve during the next inter-inspection period. Because there is separate action chosen for each satellite in the constellation and because the satellites are stochastically independent, each satellite's MDP transition probability matrix can be viewed separately.

Table 4.15 Sample action set for state s_{11}, example 2.

Action Set	*Action*	*Definition*
$\mathcal{A}_{11}$	$a_{11,1}$	do nothing, do nothing, do nothing
	$a_{11,2}$	do nothing, do nothing, repair
	$a_{11,3}$	do nothing, do nothing, replace
	$a_{11,4}$	do nothing, repair, do nothing
	$a_{11,5}$	do nothing, repair, repair
	$a_{11,6}$	do nothing, repair, replace
	$a_{11,7}$	do nothing, replace, do nothing
	$a_{11,8}$	do nothing, replace, repair
	$a_{11,9}$	do nothing, replace, replace
	$a_{11,19}$	replace, do nothing, do nothing
	$a_{11,20}$	replace, do nothing, repair
	$a_{11,21}$	replace, do nothing, replace
	$a_{11,22}$	replace, repair, do nothing
	$a_{11,23}$	replace, repair, repair
	$a_{11,24}$	replace, repair, replace
	$a_{11,25}$	replace, replace, do nothing
	$a_{11,26}$	replace, replace, repair
	$a_{11,27}$	replace, replace, replace

Table 4.16 Repair and replacement probabilities for each function, example 2.

Function	H_m (*Repair Probability*)	G_m (*Replace Probability*)
Function 1	0.95	0.975
Function 2	0.96	0.94
Function 3	0.97	0.98

Because there are three possible actions for each satellite, each satellite will evolve stochastically according to one of three transition probability matrices each decision epoch. If action 1 (do nothing) is chosen for satellites 1 or 2, the satellite will evolve according to the following transition probability matrix for the next inter-inspection interval:

$$\mathbf{P} = \begin{bmatrix} R_1R_2 & R_1F_2 & F_1R_2 & F_1F_2 \\ 0 & R_1 & 0 & F_1 \\ 0 & 0 & R_2 & F_2 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{4.22}$$

If action 1 (do nothing) is chosen for satellite 3, the satellite will evolve according to the following transition probability matrix for the next inter-inspection interval:

$$\mathbf{P} = \begin{bmatrix} R_1R_2R_3 & R_1R_2F_3 & R_1F_2R_3 & R_1F_2F_3 & F_1R_2R_3 & F_1R_2F_3 & F_1F_2R_3 & F_1F_2F_3 \\ 0 & R_1R_2 & 0 & R_1F_2 & 0 & F_1R_2 & 0 & F_1F_2 \\ 0 & 0 & R_1R_3 & R_1F_3 & 0 & 0 & F_1R_3 & F_1F_3 \\ 0 & 0 & 0 & R_1 & 0 & 0 & 0 & F_1 \\ 0 & 0 & 0 & 0 & R_2R_3 & R_2F_3 & F_2R_3 & F_2F_3 \\ 0 & 0 & 0 & 0 & 0 & R_2 & 0 & F_2 \\ 0 & 0 & 0 & 0 & 0 & 0 & R_3 & F_3 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{4.23}$$

Substituting the numerical values into Equations (4.22) and (4.23) yields

$$\mathbf{P} = \begin{bmatrix} 0.91113 & 0.04444 & 0.04237 & 0.00207 \\ 0 & 0.95556 & 0 & 0.04444 \\ 0 & 0 & 0.95350 & 0.04650 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.24}$$

and

$$\mathbf{P} = \begin{bmatrix} 0.87675 & 0.03438 & 0.04276 & 0.00168 & 0.04077 & 0.00160 & 0.00199 & 0.00008 \\ 0 & 0.91113 & 0 & 0.04444 & 0 & 0.04237 & 0 & 0.00207 \\ 0 & 0 & 0.91951 & 0.03605 & 0 & 0 & 0.04276 & 0.00168 \\ 0 & 0 & 0 & 0.95556 & 0 & 0 & 0 & 0.04444 \\ 0 & 0 & 0 & 0 & 0.91752 & 0.03598 & 0.04475 & 0.00175 \\ 0 & 0 & 0 & 0 & 0 & 0.95350 & 0 & 0.04650 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.96227 & 0.03773 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1.00000 \end{bmatrix}. \tag{4.25}$$

If action 2 (on-orbit repair) is chosen for satellites 1 or 2, the satellite will evolve according to the following transition probability matrix for the next inter-inspection

interval:

$$\mathbf{P} = \begin{bmatrix} N/A & N/A & N/A & N/A \\ R_1H_2 & R_1\bar{H}_2 & F_1H_2 & F_1\bar{H}_2 \\ H_1R_2 & H_1F_2 & \bar{H}_1R_2 & \bar{H}_1F_2 \\ H_1H_2 & H_1\bar{H}_2 & \bar{H}_1H_2 & \bar{H}_1\bar{H}_2 \end{bmatrix}. \tag{4.26}$$

If action 2 (on-orbit repair) is chosen for satellite 3, the satellite will evolve according to the following transition probability matrix for the next inter-inspection interval:

$$\mathbf{P} = \begin{bmatrix} N/A & N/A & N/A & N/A & N/A & N/A & N/A & N/A \\ R_1R_2H_3 & R_1R_2\bar{H}_3 & R_1F_2H_3 & R_1F_2\bar{H}_3 & F_1R_2H_3 & F_1R_2\bar{H}_3 & F_1F_2H_3 & F_1F_2\bar{H}_3 \\ R_1H_2R_3 & R_1H_2F_3 & R_1\bar{H}_2R_3 & R_1\bar{H}_2F_3 & F_1H_2R_3 & F_1\bar{H}_2F_3 & F_1\bar{H}_2R_3 & F_1\bar{H}_2F_3 \\ R_1H_2H_3 & R_1H_2\bar{H}_3 & R_1H_2\bar{H}_3 & R_1\bar{H}_2\bar{H}_3 & F_1H_2H_3 & F_1H_2\bar{H}_3 & F_1\bar{H}_2H_3 & F_1\bar{H}_2\bar{H}_3 \\ H_1R_2R_3 & H_1R_2F_3 & \bar{H}_1F_2R_3 & H_1F_2F_3 & \bar{H}_1R_2R_3 & \bar{H}_1R_2F_3 & \bar{H}_1F_2R_3 & \bar{H}_1F_2F_3 \\ H_1R_2H_3 & H_1R_2\bar{H}_3 & H_1F_2H_3 & H_1F_2\bar{H}_3 & \bar{H}_1R_2H_3 & \bar{H}_1R_2\bar{H}_3 & \bar{H}_1F_2H_3 & \bar{H}_1F_2\bar{H}_3 \\ H_1H_2R_3 & H_1H_2F_3 & H_1\bar{H}_2R_3 & H_1\bar{H}_2F_3 & \bar{H}_1H_2R_3 & \bar{H}_1H_2F_3 & \bar{H}_1\bar{H}_2R_3 & \bar{H}_1\bar{H}_2F_3 \\ H_1H_2H_3 & H_1H_2\bar{H}_3 & H_1\bar{H}_2H_3 & H_1\bar{H}_2\bar{H}_3 & \bar{H}_1H_2H_3 & \bar{H}_1H_2\bar{H}_3 & \bar{H}_1\bar{H}_2H_3 & \bar{H}_1\bar{H}_2\bar{H}_3 \end{bmatrix}. \tag{4.27}$$

Substituting the numerical values into Equations (4.26) and (4.27) yields

$$\mathbf{P} = \begin{bmatrix} N/A & N/A & N/A & N/A \\ 0.91734 & 0.03822 & 0.04266 & 0.00178 \\ 0.90582 & 0.04418 & 0.04767 & 0.00233 \\ 0.91200 & 0.03800 & 0.04800 & 0.00200 \end{bmatrix} \tag{4.28}$$

and

$$\mathbf{P} = \begin{bmatrix} N/A & N/A & N/A & N/A & N/A & N/A & N/A & N/A \\ 0.88379 & 0.02733 & 0.04310 & 0.00133 & 0.04110 & 0.00127 & 0.00200 & 0.00006 \\ 0.88273 & 0.03461 & 0.03678 & 0.00144 & 0.04105 & 0.00161 & 0.00171 & 0.00007 \\ 0.88982 & 0.02752 & 0.03708 & 0.00115 & 0.04138 & 0.00128 & 0.00172 & 0.00005 \\ 0.87164 & 0.03418 & 0.04251 & 0.00167 & 0.04588 & 0.00180 & 0.00224 & 0.00009 \\ 0.87865 & 0.02717 & 0.04285 & 0.00133 & 0.04624 & 0.00143 & 0.00226 & 0.00007 \\ 0.87759 & 0.03441 & 0.03657 & 0.00143 & 0.04619 & 0.00181 & 0.00192 & 0.00008 \\ 0.88464 & 0.02736 & 0.03686 & 0.00114 & 0.04656 & 0.00144 & 0.00194 & 0.00006 \end{bmatrix}. \tag{4.29}$$

If action 3 (replace) is chosen for satellites 1 or 2, the satellite will evolve according to the following transition probability matrix for the next inter-inspection interval:

$$\mathbf{P} = \begin{bmatrix} G_1G_2 & G_1\bar{G}_2 & \bar{G}_1H_2 & \bar{G}_1\bar{G}_2 \\ G_1G_2 & G_1\bar{G}_2 & \bar{G}_1H_2 & \bar{G}_1\bar{G}_2 \\ G_1G_2 & G_1\bar{G}_2 & \bar{G}_1H_2 & \bar{G}_1\bar{G}_2 \\ G_1G_2 & G_1\bar{G}_2 & \bar{G}_1H_2 & \bar{G}_1\bar{G}_2 \end{bmatrix}. \tag{4.30}$$

If action 3 (replace) is chosen for satellite 3, the satellite will evolve according to the following transition probability matrix for the next inter-inspection interval:

$$\mathbf{P} = \begin{bmatrix} G_1G_2G_3 & G_1G_2\bar{G}_3 & G_1\bar{G}_2G_3 & G_1\bar{G}_2\bar{G}_3 & \bar{G}_1G_2G_3 & \bar{G}_1G_2\bar{G}_3 & \bar{G}_1\bar{G}_2G_3 & \bar{G}_1\bar{G}_2\bar{G}_3 \\ G_1G_2G_3 & G_1G_2\bar{G}_3 & G_1\bar{G}_2G_3 & G_1\bar{G}_2\bar{G}_3 & \bar{G}_1G_2G_3 & \bar{G}_1G_2\bar{G}_3 & \bar{G}_1\bar{G}_2G_3 & \bar{G}_1\bar{G}_2\bar{G}_3 \\ G_1G_2G_3 & G_1G_2\bar{G}_3 & G_1\bar{G}_2G_3 & G_1\bar{G}_2\bar{G}_3 & \bar{G}_1G_2G_3 & \bar{G}_1G_2\bar{G}_3 & \bar{G}_1\bar{G}_2G_3 & \bar{G}_1\bar{G}_2\bar{G}_3 \\ G_1G_2G_3 & G_1G_2\bar{G}_3 & G_1\bar{G}_2G_3 & G_1\bar{G}_2\bar{G}_3 & \bar{G}_1G_2G_3 & \bar{G}_1G_2\bar{G}_3 & \bar{G}_1\bar{G}_2G_3 & \bar{G}_1\bar{G}_2\bar{G}_3 \\ G_1G_2G_3 & G_1G_2\bar{G}_3 & G_1\bar{G}_2G_3 & G_1\bar{G}_2\bar{G}_3 & \bar{G}_1G_2G_3 & \bar{G}_1G_2\bar{G}_3 & \bar{G}_1\bar{G}_2G_3 & \bar{G}_1\bar{G}_2\bar{G}_3 \\ G_1G_2G_3 & G_1G_2\bar{G}_3 & G_1\bar{G}_2G_3 & G_1\bar{G}_2\bar{G}_3 & \bar{G}_1G_2G_3 & \bar{G}_1G_2\bar{G}_3 & \bar{G}_1\bar{G}_2G_3 & \bar{G}_1\bar{G}_2\bar{G}_3 \\ G_1G_2G_3 & G_1G_2\bar{G}_3 & G_1\bar{G}_2G_3 & G_1\bar{G}_2\bar{G}_3 & \bar{G}_1G_2G_3 & \bar{G}_1G_2\bar{G}_3 & \bar{G}_1\bar{G}_2G_3 & \bar{G}_1\bar{G}_2\bar{G}_3 \\ G_1G_2G_3 & G_1G_2\bar{G}_3 & G_1\bar{G}_2G_3 & G_1\bar{G}_2\bar{G}_3 & \bar{G}_1G_2G_3 & \bar{G}_1G_2\bar{G}_3 & \bar{G}_1\bar{G}_2G_3 & \bar{G}_1\bar{G}_2\bar{G}_3 \end{bmatrix}. \tag{4.31}$$

Substituting the numerical values into Equations (4.30) and (4.31) yields

$$\mathbf{P} = \begin{bmatrix} 0.91650 & 0.05850 & 0.02350 & 0.00150 \\ 0.91650 & 0.05850 & 0.02350 & 0.00150 \\ 0.91650 & 0.05850 & 0.02350 & 0.00150 \\ 0.91650 & 0.05850 & 0.02350 & 0.00150 \end{bmatrix} \tag{4.32}$$

and

$$\mathbf{P} = \begin{bmatrix} 0.89817 & 0.01833 & 0.05733 & 0.00117 & 0.02303 & 0.00047 & 0.00147 & 0.00003 \\ 0.89817 & 0.01833 & 0.05733 & 0.00117 & 0.02303 & 0.00047 & 0.00147 & 0.00003 \\ 0.89817 & 0.01833 & 0.05733 & 0.00117 & 0.02303 & 0.00047 & 0.00147 & 0.00003 \\ 0.89817 & 0.01833 & 0.05733 & 0.00117 & 0.02303 & 0.00047 & 0.00147 & 0.00003 \\ 0.89817 & 0.01833 & 0.05733 & 0.00117 & 0.02303 & 0.00047 & 0.00147 & 0.00003 \\ 0.89817 & 0.01833 & 0.05733 & 0.00117 & 0.02303 & 0.00047 & 0.00147 & 0.00003 \\ 0.89817 & 0.01833 & 0.05733 & 0.00117 & 0.02303 & 0.00047 & 0.00147 & 0.00003 \\ 0.89817 & 0.01833 & 0.05733 & 0.00117 & 0.02303 & 0.00047 & 0.00147 & 0.00003 \end{bmatrix}. \tag{4.33}$$

Knowledge of each satellite's individual transition probabilities is sufficient to compute the overall constellation's transition probabilities. The overall transition probabilities are computed using Equation (3.24) with $K = 3$:

$$p_n\left\{j|s, \mathbf{a}_{s,i}\right\} = \prod_{k=1}^{3} p_0^{(k)}\left\{j^{(k)}|s^{(k)}, a_{s^{(k)},i}\right\},\ n = 1, 2, \ldots, N-1. \tag{4.34}$$

The constellation transition probability matrices are too large to display here. There are a total of 27 actions and 128 states for the constellation. Therefore, each action produces a transition probability matrix that has 128 rows and columns. This means that there are a total of $128 \times 27 \times 128 = 442,368$ transition probabilities for the three satellite constellation MDP example. Table 4.14 can be referenced as an example of transition probabilities when the action is 'do nothing' for all satellites.

The cost to perform on-orbit repairs for the multiple satellite model is composed of the cost of the space vehicle, and the function specific maintenance costs. The cost to perform a replacement is composed of the the cost of the unit cost of the replacement satellite plus any function upgrades. These costs are summarized in Table 4.17.

Table 4.17 Notional on-orbit repair and replacement costs (millions), example 2.

Cost	*Satellite 1*	*Satellite 2*	*Satellite 3*
Space Vehicle	\$450	\$450	\$450
Function 1 Repair	\$35	\$35	\$35
Function 2 Repair	\$15	\$15	\$15
Function 3 Repair	N/A	N/A	\$20
Unit Cost	\$470	\$470	\$470
Function Upgrade Cost	\$30	\$30	\$50

Suppose that for the satellite constellation to be considered fully capable, two satellites must have function 1 operational, and two satellites must have function 2 operational. In other words, function 1 must be non-operational on two or more

satellites before a penalty cost is assessed. The same can be said about function 2. Also, suppose that function 3 improves performance of the constellation, however, the mission can be completed successfully when the function is non-operational. In other words, a small penalty cost is assessed when function 3 fails. Table 4.18 summarizes the notional penalty costs for this scenario.

Table 4.18 Notional penalty costs assigned (millions), example 2.

Number Operational	*Function 1*	*Function 2*	*Function 3*
3	\$0	\$0	N/A
2	\$0	\$0	N/A
1	\$500	\$450	\$0
0	\$650	\$700	\$200

The rewards for the three satellite model are found using Equation (3.28) with $K = 3$ and $|\mathcal{S}| = 128$. That is

$$r_n(s,a) = \sum_{k=1}^{3} r^{(k)}(a) + \sum_{j=1}^{128} C_p(j) p_n(j|s,a)$$

where $p_n(j|s,a)$ is found with Equation (4.34). The enumerated list of rewards is too large to display here, however, Table 4.19 gives some example rewards.

Table 4.19 Expected rewards for the first $N - 1$ decision epochs, example 2.

State	*Action*	*Reward*	*Expected Value Formula*	*Value (millions)*
s_1	$(1,1,1)$	$r_n(s_1, a_{1,1})$	$0 + \sum_{j \in \mathcal{S}} C_p(j) p\{j \vert s_1, a_{1,1}\}$	\$13.2877
s_1	$(1,1,3)$	$r_n(s_1, a_{1,3})$	$\$520 + \sum_{j \in \mathcal{S}} C_p(j) p\{j \vert s_1, a_{1,3}\}$	\$529.4563
s_{128}	$(2,1,2)$	$r_n(s_{128}, a_{128,11})$	$\$500 + \$520 + \sum_{j \in \mathcal{S}} C_p(j) p\{j \vert s_{128}, a_{128,11}\}$	\$1110.805
s_{128}	$(1,3,1)$	$r_n(s_{128}, a_{128,7})$	$\$500 + \sum_{j \in \mathcal{S}} C_p(j) p\{j \vert s_{128}, a_{128,7}\}$	\$1668.75

Because these rewards look into the next time interval to compute the expected rewards, and because there are no decisions made at the final decision epoch, $N = 20$, there is no terminal reward so that $r_{20}(s_1) = r_{20}(s_2) = \ldots = r_{20}(s_{128}) = 0$.

4.3.3 Three-Satellite Optimality Results

The three-satellite example was solved by means of the backward induction algorithm coded in MATLAB®. The optimal policy found using this method is presented in Tables 4.20, 4.21, and 4.22. This policy lists the optimal decision rule for each decision epoch. Each decision rule gives the optimal action to take for every possible state the system may be found in at that decision epoch. For example, if the constellation is found to be in state s_{26} (representing the state vector $(1, 4, 2)$) after 2 years ($n = 8$) then the optimal action to take is action 8 (representing the action vector $(1, 3, 2)$). In other words, suppose that at inspection epoch 8 the following observation of the constellation is made:

- Satellite 1 is fully capable,
- satellite 2 is fully non-capable, and
- satellite 3 has functions 1 and 2 operational and function 3 non-operational.

Then the actions for each satellite that will produce the minimum total expected loss over the rest of the constellation's lifetime are as follows:

- Do nothing for satellite 1;
- replace satellite 2; and
- perform an on-orbit repair to satellite 3.

The total expected loss that results from the optimal policy depends on the initial state of the constellation. Table 4.23 gives the optimal loss for every possible initial state.

Table 4.20 Three-satellite optimal policy (states $s_1 - s_{42}$), example 2.

State	Decision Epoch (n)																		
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
s_1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
s_2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	1	1
s_3	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
s_4	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	1	1
s_5	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
s_6	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	1	1
s_7	2	2	2	2	2	2	2	2	2	2	2	2	2	2	1	1	1	1	1
s_8	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	1
s_9	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
s_{10}	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	1	1
s_{11}	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	1
s_{12}	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
s_{13}	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
s_{14}	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	1	1
s_{15}	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	1
s_{16}	3	3	3	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
s_{17}	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
s_{18}	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	1	1
s_{19}	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
s_{20}	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	1	1
s_{21}	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	1
s_{22}	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	2
s_{23}	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	1
s_{24}	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
s_{25}	7	7	7	7	7	7	7	7	7	7	7	7	7	7	1	1	1	1	1
s_{26}	8	8	8	8	8	8	8	8	8	8	8	8	8	8	2	2	2	1	1
s_{27}	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	1
s_{28}	8	8	8	8	8	8	8	8	8	8	8	8	8	8	2	2	2	2	2
s_{29}	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7
s_{30}	8	8	8	8	8	8	8	8	8	8	8	8	8	8	3	3	3	2	2
s_{31}	8	8	8	8	8	8	8	8	8	8	8	8	8	8	7	7	7	7	7
s_{32}	9	9	9	9	9	9	9	9	9	9	9	9	9	9	3	3	3	3	3
s_{33}	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
s_{34}	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	1	1
s_{35}	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	1
s_{36}	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
s_{37}	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
s_{38}	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	1	1
s_{39}	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	1
s_{40}	3	3	3	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
s_{41}	10	10	4	10	4	4	10	10	10	10	4	4	4	4	4	4	4	4	1
s_{42}	11	11	11	5	5	5	5	11	11	11	5	5	5	5	5	5	5	4	1

Table 4.21 Three-satellite optimal policy (states $s_{43} - s_{84}$), example 2.

State	Decision Epoch (n)																	
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
s_{43}	13	13	13	13	13	13	13	13	13	13	13	13	13	13	13	13	13	13	1
s_{44}	11	11	5	5	5	5	5	5	5	5	11	5	5	5	5	5	5	5	1
s_{45}	10	10	10	10	4	4	10	4	10	4	4	4	10	4	4	4	4	4	1
s_{46}	11	11	11	11	5	5	5	5	11	5	5	5	5	5	5	5	5	4	1
s_{47}	11	11	11	5	5	5	5	5	11	11	5	5	5	5	5	11	5	5	1
s_{48}	12	12	11	11	5	5	5	11	5	11	5	5	5	5	5	5	5	5	1
s_{49}	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
s_{50}	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	1	1
s_{51}	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	2	1
s_{52}	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
s_{53}	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	1
s_{54}	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	2
s_{55}	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
s_{56}	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
s_{57}	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	1
s_{58}	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	4	1
s_{59}	13	13	13	13	13	13	13	13	13	13	13	13	13	13	13	13	13	13	1
s_{60}	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1
s_{61}	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7
s_{62}	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	7	7
s_{63}	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	7
s_{64}	6	6	6	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	3
s_{65}	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
s_{66}	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	1	1
s_{67}	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
s_{68}	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	1	1
s_{69}	19	19	19	19	19	19	19	19	19	19	19	19	19	19	19	19	19	19	1
s_{70}	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	2
s_{71}	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	1
s_{72}	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
s_{73}	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
s_{74}	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	1	1
s_{75}	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	2	1
s_{76}	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
s_{77}	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	1
s_{78}	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	2
s_{79}	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
s_{80}	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
s_{81}	7	7	7	19	7	7	7	7	7	7	19	7	7	7	7	7	7	7	1
s_{82}	8	8	8	8	20	8	8	8	8	8	8	8	8	8	8	8	8	19	1
s_{83}	4	10	10	4	4	4	4	4	10	4	4	4	4	4	4	4	4	4	1
s_{84}	8	8	8	8	20	8	8	8	8	8	8	20	8	8	8	8	8	4	1

Table 4.22 Three-satellite optimal policy (states $s_{85} - s_{128}$), example 2.

State	Decision Epoch (n)																	
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
s_{85}	25	25	25	25	25	25	25	25	25	25	25	25	25	25	25	25	25	25	1
s_{86}	9	9	9	9	9	21	9	9	9	9	9	9	9	9	9	9	9	9	1
s_{87}	9	9	9	9	9	21	9	9	9	9	9	9	9	9	9	9	9	9	1
s_{88}	9	9	9	9	9	21	9	9	9	9	9	9	9	9	9	9	9	9	1
s_{89}	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7
s_{90}	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	7	7
s_{91}	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7
s_{92}	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	7	7
s_{93}	25	25	25	25	25	25	25	25	25	25	25	25	25	25	25	25	25	25	1
s_{94}	9	9	9	9	9	9	9	9	9	9	9	9	9	9	9	9	9	9	1
s_{95}	9	9	9	9	9	9	9	9	9	9	9	9	9	9	9	9	9	9	8
s_{96}	9	9	9	9	9	9	9	9	9	9	9	9	9	9	9	9	9	9	9
s_{97}	19	19	19	19	19	19	19	19	19	19	19	19	19	19	1	1	1	1	1
s_{98}	20	20	20	20	20	20	20	20	20	20	20	20	20	20	2	2	2	1	1
s_{99}	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	1
s_{100}	20	20	20	20	20	20	20	20	20	20	20	20	20	20	2	2	2	2	2
s_{101}	19	19	19	19	19	19	19	19	19	19	19	19	19	19	19	19	19	19	19
s_{102}	20	20	20	20	20	20	20	20	20	20	20	20	20	20	3	3	3	2	2
s_{103}	20	20	20	20	20	20	20	20	20	20	20	20	20	20	19	19	19	19	19
s_{104}	21	21	21	21	21	21	21	21	21	21	21	21	21	21	3	3	3	3	3
s_{105}	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	1
s_{106}	11	11	11	11	11	11	11	11	11	11	11	11	11	11	11	11	11	10	1
s_{107}	13	13	13	13	13	13	13	13	13	13	13	13	13	13	13	13	13	13	1
s_{108}	11	11	11	11	11	11	11	11	11	11	11	11	11	11	11	11	11	11	1
s_{109}	19	19	19	19	19	19	19	19	19	19	19	19	19	19	19	19	19	19	19
s_{110}	11	11	11	11	11	11	11	11	11	11	11	11	11	11	11	11	11	19	19
s_{111}	11	11	11	11	11	11	11	11	11	11	11	11	11	11	11	11	11	11	19
s_{112}	12	12	12	11	11	11	11	11	11	11	11	11	11	11	11	11	11	11	3
s_{113}	19	19	19	19	19	19	19	19	19	19	19	19	19	19	19	19	19	19	19
s_{114}	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	19	19
s_{115}	19	19	19	19	19	19	19	19	19	19	19	19	19	19	19	19	19	19	19
s_{116}	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	19	19
s_{117}	25	25	25	25	25	25	25	25	25	25	25	25	25	25	25	25	25	25	1
s_{118}	21	21	21	21	21	21	21	21	21	21	21	21	21	21	21	21	21	21	1
s_{119}	21	21	21	21	21	21	21	21	21	21	21	21	21	21	21	21	21	21	20
s_{120}	21	21	21	21	21	21	21	21	21	21	21	21	21	21	21	21	21	21	21
s_{121}	25	25	25	25	25	25	25	25	25	25	25	25	25	25	7	7	7	7	7
s_{122}	26	26	26	26	26	26	26	26	26	26	26	26	26	26	8	8	8	7	7
s_{123}	13	13	13	13	13	13	13	13	13	13	13	13	13	13	13	13	13	13	7
s_{124}	26	26	26	26	26	26	26	26	26	26	26	26	26	26	8	8	8	8	8
s_{125}	25	25	25	25	25	25	25	25	25	25	25	25	25	25	25	25	25	25	25
s_{126}	26	26	26	26	26	26	26	26	26	26	26	26	26	26	9	9	9	9	8
s_{127}	26	26	26	26	26	26	26	26	26	26	26	26	26	26	25	25	25	25	25
s_{128}	27	27	27	27	27	27	27	27	27	27	27	27	27	27	9	9	9	9	9

Table 4.23 Three-satellite minimum expected total loss for every possible initial state (*millions*).

State	*Loss*	*State*	*Loss*	*State*	*Loss*	*State*	*Loss*
s_1	2130.4974	s_{33}	2566.41431	s_{65}	2598.18333	s_{97}	2627.26077
s_2	2595.59829	s_{34}	3031.51996	s_{66}	3063.28906	s_{98}	3092.36163
s_3	2490.08459	s_{35}	2949.36661	s_{67}	2947.86501	s_{99}	2987.04673
s_4	2608.12886	s_{36}	3041.30435	s_{68}	3075.88774	s_{100}	3104.85193
s_5	2521.32085	s_{37}	2945.95808	s_{69}	3008.96892	s_{101}	3008.96892
s_6	2632.8883	s_{38}	3068.74375	s_{70}	3095.01841	s_{102}	3129.59855
s_7	2630.329	s_{39}	3063.43577	s_{71}	3095.01841	s_{103}	3126.99905
s_8	2636.42943	s_{40}	3078.29776	s_{72}	3095.01841	s_{104}	3133.46205
s_9	2566.41431	s_{41}	3025.68798	s_{73}	3019.62094	s_{105}	3063.34134
s_{10}	3031.51996	s_{42}	3490.79364	s_{74}	3484.74032	s_{106}	3528.44708
s_{11}	2949.36661	s_{43}	3408.64026	s_{75}	3407.16404	s_{107}	3446.32048
s_{12}	3041.30435	s_{44}	3500.5696	s_{76}	3494.51039	s_{108}	3538.22295
s_{13}	2945.95808	s_{45}	3405.24506	s_{77}	3436.24471	s_{109}	3439.46252
s_{14}	3068.74375	s_{46}	3528.0175	s_{78}	3522.31108	s_{110}	3565.68637
s_{15}	3063.43577	s_{47}	3522.7011	s_{79}	3519.27779	s_{111}	3560.36981
s_{16}	3078.29776	s_{48}	3537.58864	s_{80}	3522.31108	s_{112}	3575.18845
s_{17}	2598.18333	s_{49}	3019.62094	s_{81}	3085.77006	s_{113}	3085.77006
s_{18}	3063.28906	s_{50}	3484.74032	s_{82}	3550.87597	s_{114}	3550.87597
s_{19}	2947.86501	s_{51}	3407.16404	s_{83}	3438.15927	s_{115}	3441.31771
s_{20}	3075.88774	s_{52}	3494.51039	s_{84}	3563.43461	s_{116}	3563.43461
s_{21}	3008.96892	s_{53}	3436.24471	s_{85}	3496.56847	s_{117}	3496.56847
s_{22}	3095.01841	s_{54}	3522.31108	s_{86}	3582.5561	s_{118}	3582.5561
s_{23}	3095.01841	s_{55}	3519.27779	s_{87}	3582.5561	s_{119}	3582.5561
s_{24}	3095.01841	s_{56}	3522.31108	s_{88}	3582.5561	s_{120}	3582.5561
s_{25}	2627.26077	s_{57}	3063.34134	s_{89}	3085.77006	s_{121}	3124.29585
s_{26}	3092.36163	s_{58}	3528.44708	s_{90}	3550.87597	s_{122}	3589.39668
s_{27}	2987.04673	s_{59}	3446.32048	s_{91}	3441.31771	s_{123}	3484.01603
s_{28}	3104.85193	s_{60}	3538.22295	s_{92}	3563.43461	s_{124}	3601.84751
s_{29}	3008.96892	s_{61}	3439.46252	s_{93}	3496.56847	s_{125}	3496.56847
s_{30}	3129.59855	s_{62}	3565.68637	s_{94}	3582.5561	s_{126}	3626.57863
s_{31}	3126.99905	s_{63}	3560.36981	s_{95}	3582.5561	s_{127}	3623.93977
s_{32}	3133.46205	s_{64}	3575.18845	s_{96}	3582.5561	s_{128}	3630.77118

4.3.4 Three-Satellite Sensitivity Analysis

As in the one-satellite example, the optimal policy and resulting optimal values are the result of a set of fixed input parameters that may or may not be accurate. Therefore, the conditions for which mixed policies are optimal are also considered for the three-satellite model. As mentioned in the MDP formulation, an on-orbit repair cost consists of the space vehicle and the function specific repair cost. For this analysis, the function specific repair costs for each satellite are assumed to be constant. However, the space vehicle cost is varied from \$10 million to \$1 billion to see its effect on the minimum total expected loss. Likewise, the unit cost for a replacement is varied from \$10 million to \$1 billion to see its effect on the minimum total expected loss. Due to the large number of states, only state s_1, fully capable, is considered for the starting state.

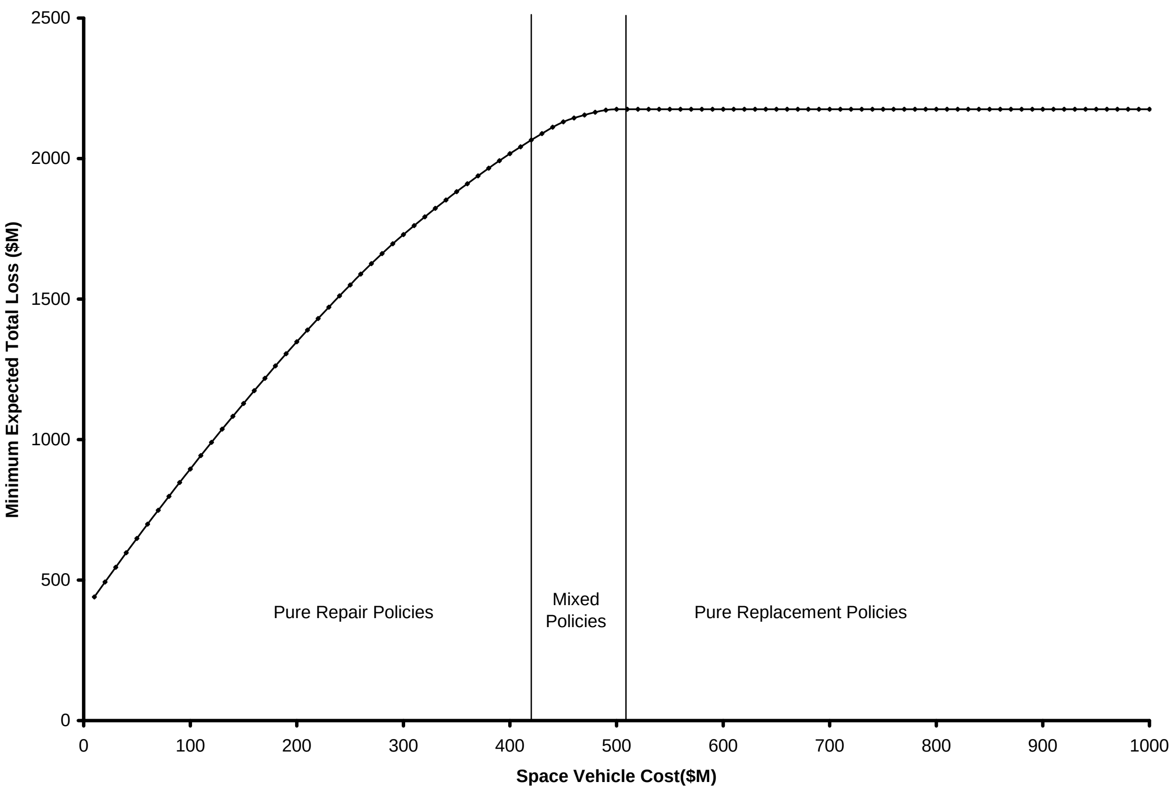

Figure 4.7 One-way sensitivity analysis of the minimum total expected loss when varying the space vehicle cost from \$10 million to \$1 billion while holding the unit cost at \$470 million, example 2.

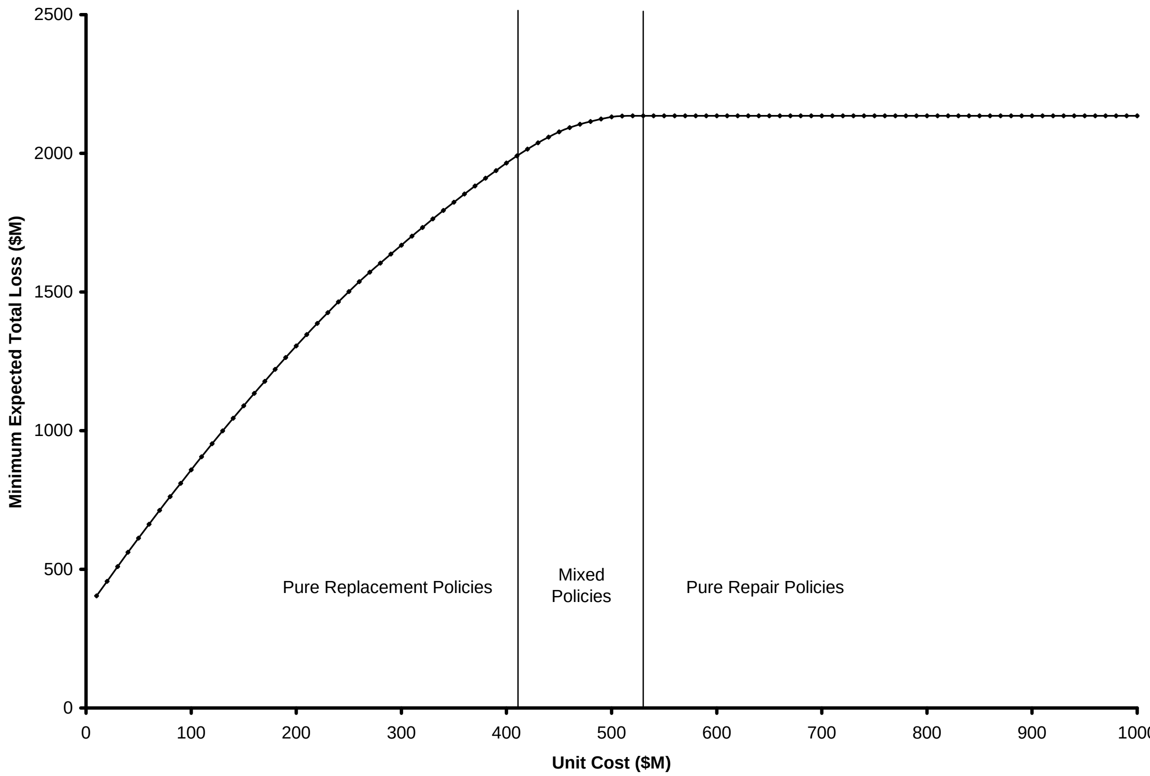

Figure 4.8 One-way sensitivity analysis of the minimum total expected loss when varying the unit cost from \$10 million to \$1 billion while holding the space vehicle cost at \$450 million, example 2.

Figure 4.7 depicts graphically the change in the minimum total expected loss caused by varying the cost of the space vehicle with all else held constant. Figure 4.8 depicts graphically the change in the minimum total expected loss caused by varying the unit cost of a satellite with all else held constant. As in the one-satellite model, the two figures illustrate that mixed policies are optimal only when the difference between on-orbit repair costs and satellite replacement costs is small. If the on-orbit repair cost is significantly larger than the replacement cost, then the optimal policy consists only of replacement actions. Likewise, if the replacement cost is significantly larger than the on-orbit repair cost, then the optimal policy consists only of on-orbit repair actions. Looking at Figure 4.7 we can see that when the unit cost of a replacement satellite is fixed at \$470 million, then mixed policies are optimal when

the space vehicle cost is between \$420 million and \$510 million, or approximately 0.9 to 1.09 of the unit cost of a replacement satellite.

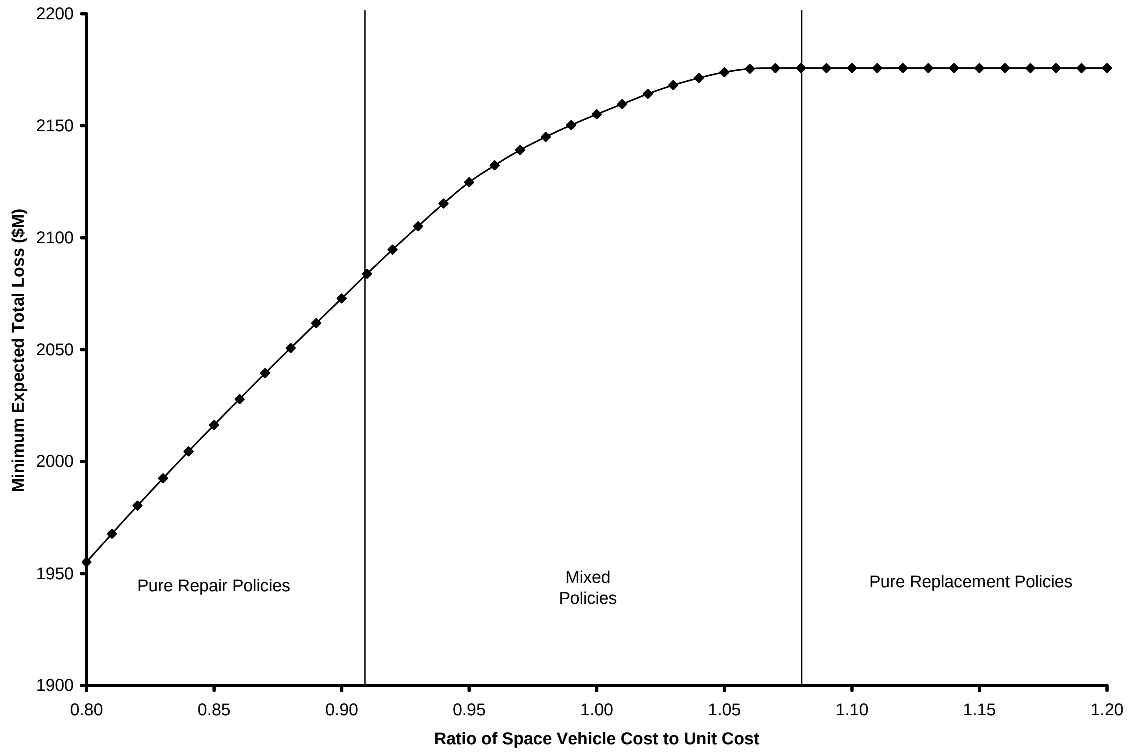

Figure 4.9 One-way sensitivity analysis of the minimum total expected loss when varying the space vehicle cost to unit replacement cost ratio from 0.85 to 1.15 while holding the unit replacement cost at \$470 million, example 2.

For further insight, another one-way analysis was done to examine the space vehicle cost to replacement cost ratios for which mixed policies are optimal when the unit replacement cost is fixed at \$470 million. As seen in Figure 4.9, mixed policies are optimal when the ratio is between 0.91 and 1.08. It is worth mentioning that under the assumptions of this analysis, on-orbit repairs appear in the optimal policy for all ratios less than 1.08.

Next, we consider a two-way sensitivity analysis to determine the maintenance cost ratios for which mixed policies are optimal. To do this, the unit replacement cost is varied from \$50 million to \$1 billion in \$50 million increments while the space

vehicle cost is varied from 0 to 120% of the unit replacement cost. The results of this analysis are depicted graphically in Figure 4.10.

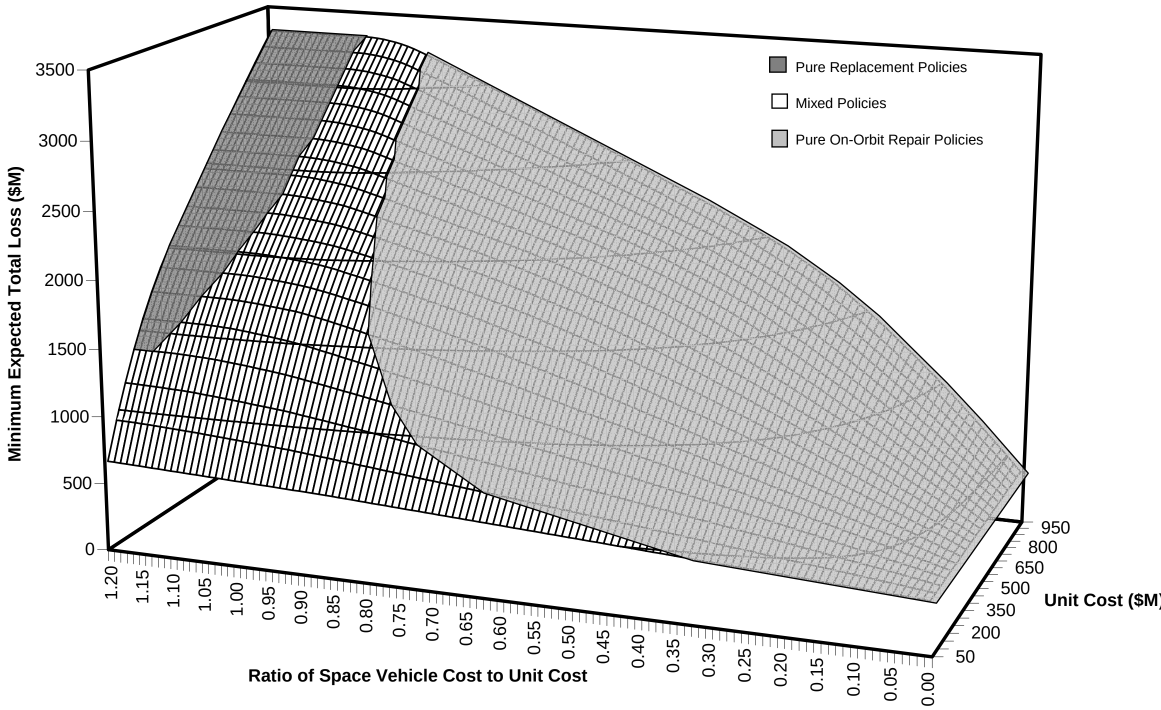

Figure 4.10 Two-way sensitivity analysis of the minimum total expected loss when varying the unit replacement cost from $50 million to $1 billion while varying the ratio of the space vehicle cost to unit replacement cost from 0.0 to 1.2, example 2.

As can be seen in Figure 4.10, the cost ratios for which mixed policies are optimal depends on the magnitude of the maintenance costs. The range of cost ratios for which mixed policies are optimal is wider for small magnitudes than for large magnitudes. Of notable interest is the fact that, under the conditions of this analysis, when the unit cost of a replacement satellite is less than $200 million, the maintenance cost ratios for which a pure replacement policy is optimal are greater than 1.20. Therefore, we performed further analysis to determine the ratios for which pure replacement policies would be optimal when the unit cost of a replacement satellite takes on the values of $150 million, $100 million, and $50 million. The results of this analysis are displayed in Table 4.24.

Table 4.24 Ratios for which pure replacement policies are optimal when the unit cost of a satellite replacement is less thatn \$200 million, example 2.

Unit Replacement Cost	*Ratio*
\$150 million	1.22
\$100 million	1.34
\$50 million	1.67

4.4 Summary

In this chapter the model described in Chapter 3 was illustrated in two numerical examples. The parameter values of each example were of a notional nature, but provided realistic scenarios. In each example, the stochastic degradation process was described first followed by an optimization problem formulation in the framework of a Markov decision process. The solution to the MDP was an easily interpretable maintenance policy which yields the minimum total expected loss over a finite planning horizon. A sensitivity analysis of the model parameters determined the conditions for which maintenance policies including on-orbit repairs would be optimal.

Chapter 5 concludes this thesis with final remarks in three areas. First, a summary of this research is presented with an emphasis on the specific contributions. Next, a description of the insights gained throughout this research is presented. Finally, suggestions are made as to the most fruitful areas of future research.

5. Conclusions and Future Research

Satellite constellations are extremely valuable assets in both the public and private sectors. Millions of customers all over the world rely on the services provided by constellations every day. However, satellites in a constellation may degrade over time resulting in a reduction in the quality of service provided. Therefore, it is desirable to determine a strategy to minimize the losses incurred when the constellation operates in a degraded state.

Currently, satellite constellation budget planners view satellite constellation maintenance as a resource allocation problem. Any analysis done to aide in the maintenance planning process is focused on determining which type of satellite acquisition would be most beneficial without exceeding a fixed budget. This method may result in extremely high (and unnecessary) expenditures. A better approach may be to determine a maintenance policy that balances the losses incurred from service degradation with the costs of maintaining a satellite constellation. This thesis addressed the satellite constellation maintenance problem within the framework of an optimal maintenance policy problem.

Optimal maintenance policies are an important area of study in operations research. Most research in this area is concerned with finding a policy of preventive maintenance which minimizes the long-run cost per unit time. The majority of these models are for single-unit systems; however, interest in the multi-unit model has been steadily increasing over the past two decades. Multi-unit models most often consider systems with increasing failure rates or systems with identical, constant failure rates. This thesis has extended the literature by presenting a methodology to determine an optimal maintenance policy for a multi-unit system in which each unit may operate in a partially degraded state. A satellite constellation is a multi-unit system composed of multi-component satellites. The satellite components are the functions defined in

Chapter 3 and are assumed to possess a constant failure rate; however, the functions are not necessarily identical.

In order to develop an optimal maintenance policy, we first studied the stochastic evolution of an unmaintained satellite constellation. Chapter 3 showed that under a minimal set of assumptions, the stochastic degradation of a satellite constellation may be modelled as a discrete-time Markov chain. The stochastic degradation model was then used to develop an optimization problem through which an optimal reactive maintenance policy could be determined. This optimization problem was formulated as a finite-horizon Markov decision process.

After the analytical model had been developed, two numerical examples were presented to illustrate its utility. The first example was a one-satellite constellation with eight possible capability levels and three possible maintenance actions. The second example was a three-satellite constellation with 128 possible capability levels and 27 possible maintenance actions. Numerical results were computed for each example using the backward induction algorithm of Puterman [32:92]. Furthermore, a sensitivity analysis was performed on the parameters of each example to determine the conditions under which on-orbit repairs would be a part of the optimal maintenance policy.

This research provides further insight into the satellite constellation maintenance problem and should serve as a building block for future research. As in any mathematical model, simplifying assumptions were made in order to ensure tractability of an analytical solution. Future research may include the relaxing of each of these assumptions to increase the applicability of the model to more complex real world problems. A discussion of these assumption relaxations follows.

This endeavor considers satellites with functions whose lifetime distributions are assumed to be mutually independent. However, some functions may actually become non-operational as a result of some other function becoming non-operational. Similarly, a redundant function may be switched to operational as a result of another

function becoming non-operational. These dependencies could be incorporated into the current model to more accurately depict the degradation of various satellites. Incorporating these dependencies could complicate the reward structure and would require knowledge of the correlation structure of the dependent variables but would not increase the cardinality of the state or action space. An example of how this relaxation could be implemented follows.

Consider two functions of a surveillance satellite - inertial stabilization (X_1) and photographic imaging (X_2). If the inertial stabilization function of a surveillance satellite becomes non-operational, then the optical lenses may become oriented in the wrong direction thereby making the photographic imaging function non-operational. This dependency could be modelled by defining the event $X_1 = 0$ as being equivalent to the event $(X_1, X_2) = (0, 0)$. In other words, if the inertial stabilization function is non-operational, the photographic imaging function is also non-operational. Incorporating dependencies with a positive correlation structure such as this will result in decreasing the cardinality of the state space.

We also assumed that the degradation of satellites within a constellation are independent. This implies that the operational status of the functions of a satellite do not affect the operational status of any functions on any other satellites. This assumption may not be accurate for some constellations such as those with satellites that act as relays. In this case, a non-operational relay channel may cause a communication channel on another satellite to become non-operational. These dependencies could be modelled in the same manner as described for dependent functions on the same satellite.

We assumed that if two or more satellites have the same function, then the failure rate of the function is identical for both satellites. In reality, if two satellites have the same function, the newer satellite may have a lower failure rate due to technological improvements that have increased the function's reliability. In such a case, this assumption can be relaxed by naming the function differently for each

satellite and assigning each a distinct failure rate. This would allow the model to be applied to more complex constellations and will have no impact on the size or computational intensity of the problem.

There is a large setup cost associated with performing an on-orbit repair due to the inaccessible nature of the space environment. Therefore, we assumed that all non-operational functions on a satellite would be repaired at the same time when the action on-orbit repair is chosen. As space maintenance technology evolves, the setup cost may decrease making it economically beneficial to consider performing on-orbit repairs to non-operational functions at different times. It may also be beneficial to consider performing on-orbit preventive maintenance to operational functions if the functions exhibit an increasing failure rate. Incorporating either of these assumption relaxations into the model would have the effect of increasing the dimensionality of the action space.

We assumed that if a function is successfully repaired or survives a replacement attempt then the function will remain operational until the next inspection epoch. This implies that the probability of an infant mortality is included in the probability of a successful maintenance attempt. An interesting extension would be to model the the period during which the function may be subject to an infant mortality separate from the probability of successful maintenance. Implementing this extension could make the model more accurately represent the real world system.

We assumed that a successful repair would return the function to new condition. That is, the function would become operational and would retain a new condition failure rate. An on-orbit repair may actually result in returning the function to the operational state, but with an increased failure rate due to stresses incurred at failure. An interesting extension would be to consider failure rates to be dependent on the number of repairs performed on the function. One method of accomplishing this could be to model each function's operational status as a bivariate process giving the status and number of failures of the function. This would greatly increase the

cardinality of the state space making the problem more computationally intensive; however, incorporating this extension would allow more complex constellations to be modelled.

Satellites often operate for a number of years before they are replaced. The replacement satellite is often the beneficiary of numerous technological improvements since the launch of the previous satellite. Therefore, satellites most likely are not replaced by identical satellites as we have assumed. Relaxing this assumption to allow satellites to be replaced by non-identical satellites would remove the Markov property from the decision problem and increase the complexity of the model. However, incorporating this change would greatly increase the model's representation of the real world.

We have considered the number of satellites to be constant throughout the planning horizon. However, if a satellite being replaced remains partially capable, it may not be removed from the constellation when the replacement satellite arrives. In other words, a satellite replacement may actually be a satellite addition. This implies that the number of satellites in a constellation may be dynamic. Therefore, an interesting extension of this thesis is to allow the number of satellites in a constellation to change over time depending on the maintenance action. One possible method of incorporating this change is to consider the stochastic evolution of a satellite constellation as a bivariate process in which one random variable accounts for the number of satellites and the other random variable describes the capability level of the constellation. Including this change would impact the model by increasing the dimensionality of the constellation's state space and increase the likelihood of a state space explosion when applied to a large constellation. However, allowing the number of satellites in the constellation to be dynamic would broaden the applicability of the model to more complex real world systems.

The assumption of instantaneous maintenance is common in the optimal maintenance literature for general degrading systems because the time to perform main-

tenance is often negligible compared to the machine operating time. However, this may not be the case for satellite constellations. It may take months or even years to execute a maintenance action for a satellite constellation (e.g., [16]). Relaxing this assumption will have the effect of making the satellite degradation level transitions non-Markovian after a decision is made to perform maintenance. This results in reducing the analytical tractability of the problem; however, relaxing these assumptions would significantly improve the applicability of the model.

This research effort considered satellites whose function lifetimes are exponentially distributed; however, it can be argued that this is not the case (e.g., [16] and [17]). Generalizing the methodology to include non-exponentially distributed lifetimes for satellite functions could prove to be the most fruitful area of improvement. If non-exponential function lifetime distributions are incorporated, the stochastic constellation degradation process may no longer be modelled as a discrete-time Markov chain. However, a much less restrictive set of assumptions may be incorporated to allow the stochastic degradation of the constellation to be modelled as a semi-Markov process. Putting the degradation process into this framework allows a stationary optimal maintenance policy to be computed using a semi-Markov decision process as described by Puterman [32:530]. The additional flexibility in such a model may allow satellite constellation program managers to plan for preventive maintenance such as preemptive satellite replacements. An example of a specific application improvement follows.

Consider a satellite constellation containing a satellite whose operations are extremely vital to national security. In other words, if this particular satellite fails, national security may be compromised. Under these conditions, it may be beneficial to preemptively maintain the high-value satellite and reduce the chance of suffering a failure. When function lifetimes are assumed to be exponentially distributed, failures are truly random events; therefore, preemptively maintaining the satellite does not reduce the probability that it fails in the next instant. As a result, the optimal policy

will call for maintenance of the satellite the first time it is observed to be in the failed state. However, if the satellite displays non-exponentially distributed function lifetimes with increasing failure rates (IFR), preventively replacing the satellite will decrease the probability of suffering a failure in the next instant.

In conclusion, optimal maintenance policies for degrading satellite constellations present a number of challenges for future research. Traditional methods of determining such policies for general degrading systems have been explored in great detail in the literature; however the inaccessibility of the environment in which satellite constellations operate makes the constellation maintenance problem unique. This thesis provided further insight into this problem by allowing each satellite in the constellation to degrade over time leaving the constellation in a finite number of partially capable states. While this is not the end-all solution to the satellite constellation maintenance problem, it serves as a stepping stone for future research that may provide the fidelity needed to implement such procedures for the analysis of both public and private sector satellite constellations.

Appendix A. One-Satellite Code

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Author: 1Lt Tim Cook
%         AFIT/ENS/GOR-05M
%         March 2005
% This program solves a Markov decision process formulated to find an
% optimal maintenance policy for a one-satellite constellation.
%
% This code assumes that the satellite will be replaced with an identical
% satellite.  It also assumes that the action set is stationary and that
% the terminal reward is 0 for all states.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Define Parameters
N = 20;  % Length of planning horizon
M = 3;   % Number of functions
% Total number of states that the satellite can possibly enter into
TotStates = 2^M;

% Number of possible actions
%   Action 1: do nothing
%   Action 2: on-orbit repair
%   Action 3: satellite replacement
actions = 3;

% Failure rates -- 1/(expected lifetime in years)*(4 quarters)
lambda1 = 1/(5.5*4);   % Failure rate of function 1
lambda2 = 1/(5.25*4);  % Failure rate of function 2
lambda3 = 1/(6.5*4);   % Failure rate of function 3

% Penalty costs
C_p = [0 -200 -500 -600 -300 -400 -400 -700];

% On orbit repair cost by state
Launch = 450; C_m = [-realmax -(Launch + 20) -(Launch + 15)
-(Launch + 35) -(Launch + 35) -(Launch + 55) -(Launch + 50)
-(Launch + 70)];

% Satellite replacement cost
C_s = -500;

% Use when an action is not feasible for that state
NA = -99999;
```

```
% Failure probabilities for each function
F = [1-exp(-lambda1) 1-exp(-lambda2) 1-exp(-lambda3)];
% Survival probabilities for each function
R = [exp(-lambda1) exp(-lambda2) exp(-lambda3)];
% Probability of a successful repair for each function
M = [.95 .96 .97];
% Probability of a successful replacement for each function
G = [.975 .94 .98];

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Define Transition Probabilities
% p(s,a,j) ==> Probability  of going to state j, given presently in
%              state s and choosing action a
p = zeros(TotStates, actions, TotStates);

p(1,1,1) = R(1)*R(2)*R(3);
p(1,1,2) = R(1)*R(2)*F(3);
p(1,1,3) = R(1)*F(2)*R(3);
p(1,1,4) = R(1)*F(2)*F(3);
p(1,1,5) = F(1)*R(2)*R(3);
p(1,1,6) = F(1)*R(2)*F(3);
p(1,1,7) = F(1)*F(2)*R(3);
p(1,1,8) = F(1)*F(2)*F(3);

% There is no need to include the transition probabilities for action 2
% while in state 2 because the action will never be chosen.  This is because
% the transition probabilities are identical to those of action 1, but the
% large setup cost is not incurred with action 1.

p(1,3,1) = G(1)*G(2)*G(3);
p(1,3,2) = G(1)*G(2)*(1-G(3));
p(1,3,3) = G(1)*(1-G(2))*G(3);
p(1,3,4) = G(1)*(1-G(2))*(1-G(3));
p(1,3,5) = (1-G(1))*G(2)*G(3);
p(1,3,6) = (1-G(1))*G(2)*(1-G(3));
p(1,3,7) = (1-G(1))*(1-G(2))*G(3);
p(1,3,8) = (1-G(1))*(1-G(2))*(1-G(3));

p(2,1,2) = R(1)*R(2);
p(2,1,4) = R(1)*F(2);
p(2,1,6) = F(1)*R(2);
p(2,1,8) = F(1)*F(2);
```

```
p(2,2,1) = R(1)*R(2)*M(3);
p(2,2,2) = R(1)*R(2)*(1-M(3));
p(2,2,3) = R(1)*F(2)*M(3);
p(2,2,4) = R(1)*F(2)*(1-M(3));
p(2,2,5) = F(1)*R(2)*M(3);
p(2,2,6) = F(1)*R(2)*(1-M(3));
p(2,2,7) = F(1)*F(2)*M(3);
p(2,2,8) = F(1)*F(2)*(1-M(3));

p(2,3,1) = G(1)*G(2)*G(3);
p(2,3,2) = G(1)*G(2)*(1-G(3));
p(2,3,3) = G(1)*(1-G(2))*G(3);
p(2,3,4) = G(1)*(1-G(2))*(1-G(3));
p(2,3,5) = (1-G(1))*G(2)*G(3);
p(2,3,6) = (1-G(1))*G(2)*(1-G(3));
p(2,3,7) = (1-G(1))*(1-G(2))*G(3);
p(2,3,8) = (1-G(1))*(1-G(2))*(1-G(3));

p(3,1,3) = R(1)*R(3);
p(3,1,4) = R(1)*F(3);
p(3,1,7) = F(1)*R(3);
p(3,1,8) = F(1)*F(3);

p(3,2,1) = R(1)*M(2)*R(3);
p(3,2,2) = R(1)*M(2)*F(3);
p(3,2,3) = R(1)*(1-M(2))*R(3);
p(3,2,4) = R(1)*(1-M(2))*F(3);
p(3,2,5) = F(1)*M(2)*R(3);
p(3,2,6) = F(1)*M(2)*F(3);
p(3,2,7) = F(1)*(1-M(2))*R(3);
p(3,2,8) = F(1)*(1-M(2))*F(3);

p(3,3,1) = G(1)*G(2)*G(3);
p(3,3,2) = G(1)*G(2)*(1-G(3));
p(3,3,3) = G(1)*(1-G(2))*G(3);
p(3,3,4) = G(1)*(1-G(2))*(1-G(3));
p(3,3,5) = (1-G(1))*G(2)*G(3);
p(3,3,6) = (1-G(1))*G(2)*(1-G(3));
p(3,3,7) = (1-G(1))*(1-G(2))*G(3);
p(3,3,8) = (1-G(1))*(1-G(2))*(1-G(3));

p(4,1,4) = R(1);
```

```
p(4,1,8) = F(1);

p(4,2,1) = R(1)*M(2)*M(3);
p(4,2,2) = R(1)*M(2)*(1-M(3));
p(4,2,3) = R(1)*(1-M(2))*M(3);
p(4,2,4) = R(1)*(1-M(2))*(1-M(3));
p(4,2,5) = F(1)*M(2)*M(3);
p(4,2,6) = F(1)*M(2)*(1-M(3));
p(4,2,7) = F(1)*(1-M(2))*M(3);
p(4,2,8) = F(1)*(1-M(2))*(1-M(3));

p(4,3,1) = G(1)*G(2)*G(3);
p(4,3,2) = G(1)*G(2)*(1-G(3));
p(4,3,3) = G(1)*(1-G(2))*G(3);
p(4,3,4) = G(1)*(1-G(2))*(1-G(3));
p(4,3,5) = (1-G(1))*G(2)*G(3);
p(4,3,6) = (1-G(1))*G(2)*(1-G(3));
p(4,3,7) = (1-G(1))*(1-G(2))*G(3);
p(4,3,8) = (1-G(1))*(1-G(2))*(1-G(3));

p(5,1,5) = R(2)*R(3);
p(5,1,6) = R(2)*F(3);
p(5,1,7) = F(2)*R(3);
p(5,1,8) = F(2)*F(3);

p(5,2,1) = M(1)*R(2)*R(3);
p(5,2,2) = M(1)*R(2)*F(3);
p(5,2,3) = M(1)*F(2)*R(3);
p(5,2,4) = M(1)*F(2)*F(3);
p(5,2,5) = (1-M(1))*R(2)*R(3);
p(5,2,6) = (1-M(1))*R(2)*F(3);
p(5,2,7) = (1-M(1))*F(2)*R(3);
p(5,2,8) = (1-M(1))*F(2)*F(3);

p(5,3,1) = G(1)*G(2)*G(3);
p(5,3,2) = G(1)*G(2)*(1-G(3));
p(5,3,3) = G(1)*(1-G(2))*G(3);
p(5,3,4) = G(1)*(1-G(2))*(1-G(3));
p(5,3,5) = (1-G(1))*G(2)*G(3);
p(5,3,6) = (1-G(1))*G(2)*(1-G(3));
p(5,3,7) = (1-G(1))*(1-G(2))*G(3);
p(5,3,8) = (1-G(1))*(1-G(2))*(1-G(3));
```

```
p(6,1,6) = R(2);
p(6,1,8) = F(2);

p(6,2,1) = M(1)*R(2)*M(3);
p(6,2,2) = M(1)*R(2)*(1-M(3));
p(6,2,3) = M(1)*F(2)*M(3);
p(6,2,4) = M(1)*F(2)*(1-M(3));
p(6,2,5) = (1-M(1))*R(2)*M(3);
p(6,2,6) = (1-M(1))*R(2)*(1-M(3));
p(6,2,7) = (1-M(1))*F(2)*M(3);
p(6,2,8) = (1-M(1))*F(2)*(1-M(3));

p(6,3,1) = G(1)*G(2)*G(3);
p(6,3,2) = G(1)*G(2)*(1-G(3));
p(6,3,3) = G(1)*(1-G(2))*G(3);
p(6,3,4) = G(1)*(1-G(2))*(1-G(3));
p(6,3,5) = (1-G(1))*G(2)*G(3);
p(6,3,6) = (1-G(1))*G(2)*(1-G(3));
p(6,3,7) = (1-G(1))*(1-G(2))*G(3);
p(6,3,8) = (1-G(1))*(1-G(2))*(1-G(3));

p(7,1,7) = R(3);
p(7,1,8) = F(3);

p(7,2,1) = M(1)*M(2)*R(3);
p(7,2,2) = M(1)*M(2)*F(3);
p(7,2,3) = M(1)*(1-M(2))*R(3);
p(7,2,4) = M(1)*(1-M(2))*F(3);
p(7,2,5) = (1-M(1))*M(2)*R(3);
p(7,2,6) = (1-M(1))*M(2)*F(3);
p(7,2,7) = (1-M(1))*(1-M(2))*R(3);
p(7,2,8) = (1-M(1))*(1-M(2))*F(3);

p(7,3,1) = G(1)*G(2)*G(3);
p(7,3,2) = G(1)*G(2)*(1-G(3));
p(7,3,3) = G(1)*(1-G(2))*G(3);
p(7,3,4) = G(1)*(1-G(2))*(1-G(3));
p(7,3,5) = (1-G(1))*G(2)*G(3);
p(7,3,6) = (1-G(1))*G(2)*(1-G(3));
p(7,3,7) = (1-G(1))*(1-G(2))*G(3);
p(7,3,8) = (1-G(1))*(1-G(2))*(1-G(3));

p(8,1,8) = 1;
```

```
p(8,2,1) = M(1)*M(2)*M(3);
p(8,2,2) = M(1)*M(2)*(1-M(3));
p(8,2,3) = M(1)*(1-M(2))*M(3);
p(8,2,4) = M(1)*(1-M(2))*(1-M(3));
p(8,2,5) = (1-M(1))*M(2)*M(3);
p(8,2,6) = (1-M(1))*M(2)*(1-M(3));
p(8,2,7) = (1-M(1))*(1-M(2))*M(3);
p(8,2,8) = (1-M(1))*(1-M(2))*(1-M(3));

p(8,3,1) = G(1)*G(2)*G(3);
p(8,3,2) = G(1)*G(2)*(1-G(3));
p(8,3,3) = G(1)*(1-G(2))*G(3);
p(8,3,4) = G(1)*(1-G(2))*(1-G(3));
p(8,3,5) = (1-G(1))*G(2)*G(3);
p(8,3,6) = (1-G(1))*G(2)*(1-G(3));
p(8,3,7) = (1-G(1))*(1-G(2))*G(3);
p(8,3,8) = (1-G(1))*(1-G(2))*(1-G(3));

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Define the Immediate Rewards - deterministic part of the reward function
% ra{a}(s) ==> Immediate reward when choosing action a in state s

ra = cell(actions);
ra{1} = zeros(TotStates);
ra{2} = zeros(TotStates);
ra{3} = zeros(TotStates);

ra{2}(1) = NA;
ra{2}(2) = C_m(2);
ra{2}(3) = C_m(3);
ra{2}(4) = C_m(4);
ra{2}(5) = C_m(5);
ra{2}(6) = C_m(6);
ra{2}(7) = C_m(7);
ra{2}(8) = C_m(8);

ra{3}(1) = C_s;
ra{3}(2) = C_s;
ra{3}(3) = C_s;
ra{3}(4) = C_s;
ra{3}(5) = C_s;
ra{3}(6) = C_s;
```

```
ra{3}(7) = C_s;
ra{3}(8) = C_s;

% Define the Expected Rewards
% r(s,a) ==> The expected reward when in state s and choosing action a.
% This is equivalent to the immediate reward plus the expected penalty cost
% over the next period.
r = zeros(TotStates,actions);

for s = 1:TotStates
    for a = 1:actions
        % Calculate the expected penalty cost over the next period - the
        % sum of the penalty costs times the probability that that cost is
        % incurred.
        Expected_Penalty = 0;
        for j = 1:TotStates
            Expected_Penalty = Expected_Penalty + p(s,a,j)*C_p(j);
        end
        % Assign the expected reward resulting from taking action a in
        % state s.
        if Expected_Penalty == 0
            r(s,a) = NA; % The action is not feasible
        else
            r(s,a) = ra{a}(s) + Expected_Penalty;
        end
    end
end

% Backward Induction Code (Directly adapted from Capt Sumter's code)

% Define the u, ustar and dstar vector
u = zeros(TotStates,actions,N);
ustar = zeros(TotStates,N);
dstar = zeros(TotStates,N);
% Note: The value of any action in the final time period has zero
% reward and therefore zero utility: ustar(s,N) = 0 for all s

n = N - 1; while n >= 1
    for s = 1:TotStates
        ustar(s,n) = -realmax;
        for a = 1:actions
            if r(s,a) ~= NA  % Infeasible action, reward not applicable
                expected_value = 0;
```

```
                for j = 1:TotStates
                    expected_value = expected_value+p(s,a,j)*ustar(j,n+1);
                end
                u(s,a,n) = r(s,a) + expected_value;
                if u(s,a,n) > ustar(s,n)
                    ustar(s,n) = u(s,a,n);
                    dstar(s,n) = a;
                end
            end
        end
    end
    n = n - 1;   % Decrement the time
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Display the results to the screen
disp('Optimal Policies');
disp(dstar);
disp('Values of ustar given the starting state');
disp(ustar);

% Open file in current directory to write results to
fid = fopen('One_Sat_Solution.txt','w');

% Write the results to the file
fprintf(fid, 'Optimal Policy'); fprintf(fid, '\n'); for i =
1:TotStates;
    for j=1:N
        fprintf(fid, '%g\t',dstar(i,j));
    end
        fprintf(fid, '\n');
end fprintf(fid, '\n Optimal Values \n'); for i = 1:TotStates;
    for j=1:N
        fprintf(fid, '%8.5f\t',ustar(i,j));
    end
        fprintf(fid, '\n');
end

% Close file
fclose(fid);
```

Appendix B. Three-Satellite Code

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Author: 1Lt Tim Cook
%         AFIT/ENS/GOR-05M
%         March 2005
% This program solves a Markov decision process formulated to find an
% optimal maintenance policy for a three satellite constellation in which
% the first and second satellite have 2 functions and the third satellite
% has 3 functions.
%
% This code assumes that a satellite will be replaced with an identical
% satellite.  It also assumes that the action set is stationary and that
% the terminal reward is 0 for all states.  This solution also assumes that
% only one satellite can be launched on a single launch vehicle, but
% multiple satellites may be replaced in one time step.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Define Parameters
N = 20;       % Length of planning horizon
M = [2 2 3];  % Number of functions desired operational in each satellite
K = 3;        % Number of Satellites

% Create a vector that keeps the number of degradation levels for each
% satellite
states = zeros(K); for i=1:K
    states(i) = 2^M(i); % Number of possible states for satellite i
end

% Define the total number of states that the entire satellite constellation
% can possibly enter into
TotStates = prod(states(:,1));

% Number of possible actions for each satellite
actions = 3;

% Failure rates -- 1/(expected lifetime in years)*(4 quarters)
lambda1 = 1/(5.5*4);   % Failure rate of function 1
lambda2 = 1/(5.25*4);  % Failure rate of function 2
lambda3 = 1/(6.5*4);   % Failure rate of function 3

% On Orbit Repair Cost by state for each satellite
C_m = cell(K);
```

```
LaunchVehicle = -450;
C_m{1} = [-realmax LaunchVehicle-15 LaunchVehicle-35 LaunchVehicle-50];
C_m{2} = [-realmax LaunchVehicle-15 LaunchVehicle-35 LaunchVehicle-50];
C_m{3} = [-realmax LaunchVehicle-20 LaunchVehicle-15 LaunchVehicle-35
  LaunchVehicle-35 LaunchVehicle-55 LaunchVehicle-50 LaunchVehicle-70];

% Satellite Replacement costs
C_s = [-500 -500 -520];

% Use when an action is not feasible for that state
NA = -99999;

% Failure probabilities for each function
F = [1-exp(-lambda1) 1-exp(-lambda2) 1-exp(-lambda3)];
% Survival probabilities for each function
R = [exp(-lambda1) exp(-lambda2) exp(-lambda3)];
% Probability of a successful repair for each function
M = [.95 .96 .97];
% Probability of a successful replacement for each function
G = [.975 .94 .98];

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Define Transition Probabilities for individual satellites
% p{k}(s,a,j) ==> Probability  for satellite k of going to state j, given
%                 presently in state s and choosing action a for that
%                 individual satellite

% Allocate memory for the transition probability matricies
% using cell arrays
p = cell(K); for i = 1:K
    p(i) = {zeros(states(i), actions, states(i))};
end

% Define the transition probabilities
p{1}(1,1,1) = R(1)*R(2);
p{1}(1,1,2) = R(1)*F(2);
p{1}(1,1,3) = F(1)*R(2);
p{1}(1,1,4) = F(1)*F(2);

% There is no need to include the transition probabilities for action 2
% while in state 2 because the action will never be chosen.  This is because
% the transition probabilities are identical to those of action 1, but the
% large setup cost is not incurred with action 1.  The same holds for the other
```

```
% two satellites.

p{1}(1,3,1) = G(1)*G(2);
p{1}(1,3,2) = G(1)*(1-G(2));
p{1}(1,3,3) = (1-G(1))*G(2);
p{1}(1,3,4) = (1-G(1))*(1-G(2));

p{1}(2,1,2) = R(1);
p{1}(2,1,4) = F(1);

p{1}(2,2,1) = R(1)*M(2);
p{1}(2,2,2) = R(1)*(1-M(2));
p{1}(2,2,3) = F(1)*M(2);
p{1}(2,2,4) = F(1)*(1-M(2));

p{1}(2,3,1) = G(1)*G(2);
p{1}(2,3,2) = G(1)*(1-G(2));
p{1}(2,3,3) = (1-G(1))*G(2);
p{1}(2,3,4) = (1-G(1))*(1-G(2));

p{1}(3,1,3) = R(2);
p{1}(3,1,4) = F(2);

p{1}(3,2,1) = M(1)*R(2);
p{1}(3,2,2) = M(1)*F(2);
p{1}(3,2,3) = (1-M(1))*R(2);
p{1}(3,2,4) = (1-M(1))*F(2);

p{1}(3,3,1) = G(1)*G(2);
p{1}(3,3,2) = G(1)*(1-G(2));
p{1}(3,3,3) = (1-G(1))*G(2);
p{1}(3,3,4) = (1-G(1))*(1-G(2));

p{1}(4,1,4) = 1;

p{1}(4,2,1) = M(1)*M(2);
p{1}(4,2,2) = M(1)*(1-M(2));
p{1}(4,2,3) = (1-M(1))*M(2);
p{1}(4,2,4) = (1-M(1))*(1-M(2));

p{1}(4,3,1) = G(1)*G(2);
p{1}(4,3,2) = G(1)*(1-G(2));
p{1}(4,3,3) = (1-G(1))*G(2);
```

```
p{1}(4,3,4) = (1-G(1))*(1-G(2));

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

p{2}(1,1,1) = R(1)*R(2);
p{2}(1,1,2) = R(1)*F(2);
p{2}(1,1,3) = F(1)*R(2);
p{2}(1,1,4) = F(1)*F(2);

p{2}(1,3,1) = G(1)*G(2);
p{2}(1,3,2) = G(1)*(1-G(2));
p{2}(1,3,3) = (1-G(1))*G(2);
p{2}(1,3,4) = (1-G(1))*(1-G(2));

p{2}(2,1,2) = R(1);
p{2}(2,1,4) = F(1);

p{2}(2,2,1) = R(1)*M(2);
p{2}(2,2,2) = R(1)*(1-M(2));
p{2}(2,2,3) = F(1)*M(2);
p{2}(2,2,4) = F(1)*(1-M(2));

p{2}(2,3,1) = G(1)*G(2);
p{2}(2,3,2) = G(1)*(1-G(2));
p{2}(2,3,3) = (1-G(1))*G(2);
p{2}(2,3,4) = (1-G(1))*(1-G(2));

p{2}(3,1,3) = R(2);
p{2}(3,1,4) = F(2);

p{2}(3,2,1) = M(1)*R(2);
p{2}(3,2,2) = M(1)*F(2);
p{2}(3,2,3) = (1-M(1))*R(2);
p{2}(3,2,4) = (1-M(1))*F(2);

p{2}(3,3,1) = G(1)*G(2);
p{2}(3,3,2) = G(1)*(1-G(2));
p{2}(3,3,3) = (1-G(1))*G(2);
p{2}(3,3,4) = (1-G(1))*(1-G(2));

p{2}(4,1,4) = 1;

p{2}(4,2,1) = M(1)*M(2);
```

```
p{2}(4,2,2) = M(1)*(1-M(2));
p{2}(4,2,3) = (1-M(1))*M(2);
p{2}(4,2,4) = (1-M(1))*(1-M(2));

p{2}(4,3,1) = G(1)*G(2);
p{2}(4,3,2) = G(1)*(1-G(2));
p{2}(4,3,3) = (1-G(1))*G(2);
p{2}(4,3,4) = (1-G(1))*(1-G(2));

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

p{3}(1,1,1) = R(1)*R(2)*R(3); p{3}(1,1,2) = R(1)*R(2)*F(3);
p{3}(1,1,3) = R(1)*F(2)*R(3); p{3}(1,1,4) = R(1)*F(2)*F(3);
p{3}(1,1,5) = F(1)*R(2)*R(3); p{3}(1,1,6) = F(1)*R(2)*F(3);
p{3}(1,1,7) = F(1)*F(2)*R(3); p{3}(1,1,8) = F(1)*F(2)*F(3);

p{3}(1,3,1) = G(1)*G(2)*G(3);
p{3}(1,3,2) = G(1)*G(2)*(1-G(3));
p{3}(1,3,3) = G(1)*(1-G(2))*G(3);
p{3}(1,3,4) = G(1)*(1-G(2))*(1-G(3));
p{3}(1,3,5) = (1-G(1))*G(2)*G(3);
p{3}(1,3,6) = (1-G(1))*G(2)*(1-G(3));
p{3}(1,3,7) = (1-G(1))*(1-G(2))*G(3);
p{3}(1,3,8) = (1-G(1))*(1-G(2))*(1-G(3));

p{3}(2,1,2) = R(1)*R(2);
p{3}(2,1,4) = R(1)*F(2);
p{3}(2,1,6) = F(1)*R(2);
p{3}(2,1,8) = F(1)*F(2);

p{3}(2,2,1) = R(1)*R(2)*M(3);
p{3}(2,2,2) = R(1)*R(2)*(1-M(3));
p{3}(2,2,3) = R(1)*F(2)*M(3);
p{3}(2,2,4) = R(1)*F(2)*(1-M(3));
p{3}(2,2,5) = F(1)*R(2)*M(3);
p{3}(2,2,6) = F(1)*R(2)*(1-M(3));
p{3}(2,2,7) = F(1)*F(2)*M(3);
p{3}(2,2,8) = F(1)*F(2)*(1-M(3));

p{3}(2,3,1) = G(1)*G(2)*G(3);
p{3}(2,3,2) = G(1)*G(2)*(1-G(3));
p{3}(2,3,3) = G(1)*(1-G(2))*G(3);
p{3}(2,3,4) = G(1)*(1-G(2))*(1-G(3));
```

```
p{3}(2,3,5) = (1-G(1))*G(2)*G(3);
p{3}(2,3,6) = (1-G(1))*G(2)*(1-G(3));
p{3}(2,3,7) = (1-G(1))*(1-G(2))*G(3);
p{3}(2,3,8) = (1-G(1))*(1-G(2))*(1-G(3));

p{3}(3,1,3) = R(1)*R(3);
p{3}(3,1,4) = R(1)*F(3);
p{3}(3,1,7) = F(1)*R(3);
p{3}(3,1,8) = F(1)*F(3);

p{3}(3,2,1) = R(1)*M(2)*R(3);
p{3}(3,2,2) = R(1)*M(2)*F(3);
p{3}(3,2,3) = R(1)*(1-M(2))*R(3);
p{3}(3,2,4) = R(1)*(1-M(2))*F(3);
p{3}(3,2,5) = F(1)*M(2)*R(3);
p{3}(3,2,6) = F(1)*M(2)*F(3);
p{3}(3,2,7) = F(1)*(1-M(2))*R(3);
p{3}(3,2,8) = F(1)*(1-M(2))*F(3);

p{3}(3,3,1) = G(1)*G(2)*G(3);
p{3}(3,3,2) = G(1)*G(2)*(1-G(3));
p{3}(3,3,3) = G(1)*(1-G(2))*G(3);
p{3}(3,3,4) = G(1)*(1-G(2))*(1-G(3));
p{3}(3,3,5) = (1-G(1))*G(2)*G(3);
p{3}(3,3,6) = (1-G(1))*G(2)*(1-G(3));
p{3}(3,3,7) = (1-G(1))*(1-G(2))*G(3);
p{3}(3,3,8) = (1-G(1))*(1-G(2))*(1-G(3));

p{3}(4,1,4) = R(1);
p{3}(4,1,8) = F(1);

p{3}(4,2,1) = R(1)*M(2)*M(3);
p{3}(4,2,2) = R(1)*M(2)*(1-M(3));
p{3}(4,2,3) = R(1)*(1-M(2))*M(3);
p{3}(4,2,4) = R(1)*(1-M(2))*(1-M(3));
p{3}(4,2,5) = F(1)*M(2)*M(3);
p{3}(4,2,6) = F(1)*M(2)*(1-M(3));
p{3}(4,2,7) = F(1)*(1-M(2))*M(3);
p{3}(4,2,8) = F(1)*(1-M(2))*(1-M(3));

p{3}(4,3,1) = G(1)*G(2)*G(3);
p{3}(4,3,2) = G(1)*G(2)*(1-G(3));
p{3}(4,3,3) = G(1)*(1-G(2))*G(3);
```

```
p{3}(4,3,4) = G(1)*(1-G(2))*(1-G(3));
p{3}(4,3,5) = (1-G(1))*G(2)*G(3);
p{3}(4,3,6) = (1-G(1))*G(2)*(1-G(3));
p{3}(4,3,7) = (1-G(1))*(1-G(2))*G(3);
p{3}(4,3,8) = (1-G(1))*(1-G(2))*(1-G(3));

p{3}(5,1,5) = R(2)*R(3);
p{3}(5,1,6) = R(2)*F(3);
p{3}(5,1,7) = F(2)*R(3);
p{3}(5,1,8) = F(2)*F(3);

p{3}(5,2,1) = M(1)*R(2)*R(3);
p{3}(5,2,2) = M(1)*R(2)*F(3);
p{3}(5,2,3) = M(1)*F(2)*R(3);
p{3}(5,2,4) = M(1)*F(2)*F(3);
p{3}(5,2,5) = (1-M(1))*R(2)*R(3);
p{3}(5,2,6) = (1-M(1))*R(2)*F(3);
p{3}(5,2,7) = (1-M(1))*F(2)*R(3);
p{3}(5,2,8) = (1-M(1))*F(2)*F(3);

p{3}(5,3,1) = G(1)*G(2)*G(3);
p{3}(5,3,2) = G(1)*G(2)*(1-G(3));
p{3}(5,3,3) = G(1)*(1-G(2))*G(3);
p{3}(5,3,4) = G(1)*(1-G(2))*(1-G(3));
p{3}(5,3,5) = (1-G(1))*G(2)*G(3);
p{3}(5,3,6) = (1-G(1))*G(2)*(1-G(3));
p{3}(5,3,7) = (1-G(1))*(1-G(2))*G(3);
p{3}(5,3,8) = (1-G(1))*(1-G(2))*(1-G(3));

p{3}(6,1,6) = R(2);
p{3}(6,1,8) = F(2);

p{3}(6,2,1) = M(1)*R(2)*M(3);
p{3}(6,2,2) = M(1)*R(2)*(1-M(3));
p{3}(6,2,3) = M(1)*F(2)*M(3);
p{3}(6,2,4) = M(1)*F(2)*(1-M(3));
p{3}(6,2,5) = (1-M(1))*R(2)*M(3);
p{3}(6,2,6) = (1-M(1))*R(2)*(1-M(3));
p{3}(6,2,7) = (1-M(1))*F(2)*M(3);
p{3}(6,2,8) = (1-M(1))*F(2)*(1-M(3));

p{3}(6,3,1) = G(1)*G(2)*G(3);
p{3}(6,3,2) = G(1)*G(2)*(1-G(3));
```

```
p{3}(6,3,3) = G(1)*(1-G(2))*G(3);
p{3}(6,3,4) = G(1)*(1-G(2))*(1-G(3));
p{3}(6,3,5) = (1-G(1))*G(2)*G(3);
p{3}(6,3,6) = (1-G(1))*G(2)*(1-G(3));
p{3}(6,3,7) = (1-G(1))*(1-G(2))*G(3);
p{3}(6,3,8) = (1-G(1))*(1-G(2))*(1-G(3));

p{3}(7,1,7) = R(3);
p{3}(7,1,8) = F(3);

p{3}(7,2,1) = M(1)*M(2)*R(3);
p{3}(7,2,2) = M(1)*M(2)*F(3);
p{3}(7,2,3) = M(1)*(1-M(2))*R(3);
p{3}(7,2,4) = M(1)*(1-M(2))*F(3);
p{3}(7,2,5) = (1-M(1))*M(2)*R(3);
p{3}(7,2,6) = (1-M(1))*M(2)*F(3);
p{3}(7,2,7) = (1-M(1))*(1-M(2))*R(3);
p{3}(7,2,8) = (1-M(1))*(1-M(2))*F(3);

p{3}(7,3,1) = G(1)*G(2)*G(3);
p{3}(7,3,2) = G(1)*G(2)*(1-G(3));
p{3}(7,3,3) = G(1)*(1-G(2))*G(3);
p{3}(7,3,4) = G(1)*(1-G(2))*(1-G(3));
p{3}(7,3,5) = (1-G(1))*G(2)*G(3);
p{3}(7,3,6) = (1-G(1))*G(2)*(1-G(3));
p{3}(7,3,7) = (1-G(1))*(1-G(2))*G(3);
p{3}(7,3,8) = (1-G(1))*(1-G(2))*(1-G(3));

p{3}(8,1,8) = 1;

p{3}(8,2,1) = M(1)*M(2)*M(3);
p{3}(8,2,2) = M(1)*M(2)*(1-M(3));
p{3}(8,2,3) = M(1)*(1-M(2))*M(3);
p{3}(8,2,4) = M(1)*(1-M(2))*(1-M(3));
p{3}(8,2,5) = (1-M(1))*M(2)*M(3);
p{3}(8,2,6) = (1-M(1))*M(2)*(1-M(3));
p{3}(8,2,7) = (1-M(1))*(1-M(2))*M(3);
p{3}(8,2,8) = (1-M(1))*(1-M(2))*(1-M(3));

p{3}(8,3,1) = G(1)*G(2)*G(3);
p{3}(8,3,2) = G(1)*G(2)*(1-G(3));
p{3}(8,3,3) = G(1)*(1-G(2))*G(3);
p{3}(8,3,4) = G(1)*(1-G(2))*(1-G(3));
```

```
p{3}(8,3,5) = (1-G(1))*G(2)*G(3);
p{3}(8,3,6) = (1-G(1))*G(2)*(1-G(3));
p{3}(8,3,7) = (1-G(1))*(1-G(2))*G(3);
p{3}(8,3,8) = (1-G(1))*(1-G(2))*(1-G(3));

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Define the state space for each satellite
Sat_s = cell(K); Sat_s{1} = [1 1
    1 0
    0 1
    0 0];
Sat_s{2} = [1 1
    1 0
    0 1
    0 0];
Sat_s{3} = [1 1 1
    1 1 0
    1 0 1
    1 0 0
    0 1 1
    0 1 0
    0 0 1
    0 0 0];

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Define the overall state space for the constellation
Constellation_s = cell(TotStates);
s1 = 1;
s2 = 1;
s3 = 1;
for i = 1:TotStates
    Constellation_s{i} = [s1 s2 s3];

    s3 = s3 + 1;
    if s3 > states(3)
        s3 = 1;
        s2 = s2 + 1;
    end
    if s2 > states(2)
        s2 = 1;
        s1 = s1 + 1;
    end
    if s1 > states(1)
```

```
        s1 = 1;
    end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Penalty Cost assigned per state
% The decision maker desires that
C_p = zeros(TotStates);
for i = 1:TotStates

    % Function 1 Penalty Cost
    s1 = Constellation_s{i}(1);
    s2 = Constellation_s{i}(2);
    s3 = Constellation_s{i}(3);
    sum = Sat_s{1}(s1,1)+Sat_s{2}(s2,1)+Sat_s{3}(s3,1);
    if sum == 3
        C_p(i) = C_p(i) + 0;
    elseif sum == 2
        C_p(i) = C_p(i) + 0;
    elseif sum == 1
        C_p(i) = C_p(i) - 500;
    elseif sum == 0
        C_p(i) = C_p(i) - 650;
    end

    % Function 2 Penalty Cost
    sum = Sat_s{1}(s1,2)+Sat_s{2}(s2,2)+Sat_s{3}(s3,2);
    if sum == 3
        C_p(i) = C_p(i) + 0;
    elseif sum == 2
        C_p(i) = C_p(i) + 0;
    elseif sum == 1
        C_p(i) = C_p(i) - 450;
    elseif sum == 0
        C_p(i) = C_p(i) - 700;
    end

    % Function 3 Penalty Cost
    if Sat_s{3}(s3,3) == 1
        C_p(i) = C_p(i) + 0;
    elseif Sat_s{3}(s3,3) == 0
        C_p(i) = C_p(i) - 200;
    end
end
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Define the overall constellation action set in a cell array
Constellation_a = cell(actions^K);
a1=1;
a2=1;
a3=1;
for i = 1:actions^K
    Constellation_a{i}=[a1 a2 a3];

    a3 = a3 + 1;
    if a3 > actions
        a2 = a2 + 1;
        a3 = 1;
    end
    if a2 > actions
        a1 = a1 + 1;
        a2 = 1;
    end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Define the overall constellation transition probabilities.  Do this by
% looping through each possibile (s,a,j) combination in the constellation
% and multiplying the individual satellite transition probabilities
% together that would lead to the overall satellite constellation
% transition.  This is where the vector notation for each state comes in
% handy.  Also, the user of this code must know clearly what state the
% index of each satellite's state and action vector represents.
s1 = 1;
s2 = 1;
s3 = 1;
a1 = 1;
a2 = 1;
a3 = 1;
j1 = 1;
j2 = 1;
j3 = 1;
Constellation_p = zeros(TotStates,actions^K,TotStates);

for s = 1:TotStates
    for a = 1:actions^K
```

```
for j = 1:TotStates

    Constellation_p(s,a,j) = p{1}(s1,a1,j1)*p{2}(s2,a2,j2)*p{3}(s3,a3,j3);

    % Eliminate some transition probabilities because the actions
    % are not feasible.  This sets the transition probabilities to
    % 0 for all infeasible actions and therefore sets the expected
    % rewards to 0 so they can be identified and then set to NA.
    if s1 == 1
        if a1 == 2
            Constellation_p(s,a,j) = 0;
        end
    end
    if s2 == 1
        if a2 == 2
            Constellation_p(s,a,j) = 0;
        end
    end
    if s3 == 1
        if a3 == 2
            Constellation_p(s,a,j) = 0;
        end
    end
    % The previous block eliminates any action to do an on orbit
    % repair to a satellite that has nothing broken.

    if s1 == states(1)
        if s2 == states(2)
            if s3 == states(3)
                if a1 == 1
                    if a2 == 1
                        if a3 == 1
                            Constellation_p(s,a,j) = 0;
                        end
                    end
                end
            end
        end
    end
    % The previous block eliminates the possibility of doing
    % nothing when all functions of all satellites are
    % non-operational.
```

```
            j3 = j3 + 1;
            if j3 > states(3)
                j3 = 1;
                j2 = j2 + 1;
            end
            if j2 > states(2)
                j2 = 1;
                j1 = j1 + 1;
            end
            if j1 > states(1)
                j1 = 1;
                a3 = a3 + 1;
            end
            if a3 > actions
                a3 = 1;
                a2 = a2 + 1;
            end
            if a2 > actions
                a2 = 1;
                a1 = a1 + 1;
            end
            if a1 > actions
                a1 = 1;
                s3 = s3 + 1;
            end
            if s3 > states(3)
                s3 = 1;
                s2 = s2 + 1;
            end
            if s2 > states(2)
                s2 = 1;
                s1 = s1 + 1;
            end
            if s1 > states(1)
                s1 = 1;
            end
        end
    end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Define the Immediate Rewards for each individual satellite
```

```
% ra(a,s) ==> Immediate reward when choosing action a while in state s
% (constellation).
% Assume that the repair costs are the same for either function on all
% satellites
ra = zeros(actions^K, TotStates);

for Cons_A = 1:actions^K
    for Cons_S = 1:TotStates
        for k = 1:K

            if Constellation_a{Cons_A}(k) == 1
                ra(Cons_A,Cons_S) = ra(Cons_A,Cons_S) + 0;
            end
            if Constellation_a{Cons_A}(k) == 2
                SatState = Constellation_s{Cons_S}(k);
                ra(Cons_A,Cons_S) = ra(Cons_A,Cons_S) + C_m{k}(SatState);
            end
            if Constellation_a{Cons_A}(k) == 3
                ra(Cons_A,Cons_S) = ra(Cons_A,Cons_S) + C_s(k);
            end
        end
    end
end


% Define the Expected Rewards
% r(s,a) ==> The expected reward when in state s and choosing action a for
% the constellation. This is equivalent to the immediate reward plus the
% expected penalty cost over the next period.
r = zeros(TotStates,actions^K);

for s = 1:TotStates
    for a = 1:actions^K

        % Calculate the expected penalty cost over the next period - the
        % sum of the penalty costs times the probability that that cost is
        % incurred.
        Expected_penalty = 0;

        for j = 1:TotStates
            Expected_penalty = Expected_penalty + Constellation_p(s,a,j)*C_p(j);
        end
```

```
        % Assign the expected reward resulting from taking action a in
        % state s.
        if Expected_penalty == 0
            r(s,a) = NA;
        else
            r(s,a) = ra(a,s) + Expected_penalty;
        end

                 disp(['r(',int2str(s),',',int2str(a),')=',num2str(r(s,a))])

    end
end

% Backward Induction Code

% Define the u, ustar and dstar vector
u = zeros(TotStates,actions,N);
ustar = zeros(TotStates,N);
dstar = zeros(TotStates,N);
% Note: The value of any action in the final time period has zero
% reward and therefore zero utility ustar(s,N) = 0 for all s
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

n = N - 1; while n >= 1
    for s = 1:TotStates  % Loop through all constellation states
        ustar(s,n) = -realmax;
        for a = 1:actions^K  % Loop through the actions
            if r(s,a) ~= NA  % Infeasible action - no reward

                expected_value = 0;
                for j = 1:TotStates
                    expected_value = expected_value + Constellation_p(s,a,j)*ustar(j,n+1);
                end % End For Loop for expected value
                u(s,a,n) = r(s,a) + expected_value;

                if u(s,a,n) > ustar(s,n)
                    ustar(s,n) = u(s,a,n);
                    dstar(s,n) = a;
                end
            end

        end
    end
```

```
    n = n - 1;   % Decrement the time
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Display the results to the screen
disp('Optimal policies');
disp(dstar);
disp('Values of ustar given the starting state');
disp(ustar);

% Open file in current directory to write results to
fid = fopen('Three_Sat_Solution.txt','w');

% Write the results to the file
fprintf(fid, 'Backward Induction run time = ');
fprintf(fid, '%8.5f\t', BackwardInductionRunTime);
fprintf(fid, '\n Optimal Policy'); fprintf(fid, '\n'); for i =
1:TotStates;
    for j=1:N
        fprintf(fid, '%g\t',dstar(i,j));
    end
    fprintf(fid, '\n');
end fprintf(fid, '\n Optimal Values \n'); for i = 1:TotStates;
    for j=1:N
        fprintf(fid, '%8.5f\t',ustar(i,j));
    end
    fprintf(fid, '\n');
end

% Close file
fclose(fid);
```

Appendix C. One-Satellite One-Way Analysis

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Author: 1Lt Tim Cook
%         AFIT/ENS/GOR-05M
%         March 2005
% This program performs a one-way sensitivity analysis on a Markov
% decision process formulated to find an optimal maintenance policy for a
% one satellite constellation.  In this file, the parameters that are
% modified are the space vehicle cost and the replacement cost.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Begin the outer for loop that performs the sensitivity analysis.  The
% first time through, the launch vehicle costs are varied while the
% replacement cost is held constant at $500 million.  The second time
% through, the replacement cost is varied while the space vehicle cost is
% held constant at $450 million.
for parameter = 1:2

    % Define Parameters
    N = 20;  % Length of planning horizon
    M = 3;   % Number of functions

    % Total number of states that the satellite can possibly enter into
    TotStates = 2^M;

    % Number of possible actions
    actions = 3;

    % Failure rates -- 1/(expected lifetime in years)*(4 quarters)
    lambda1 = 1/(5.5*4);   % Failure rate of function 1
    lambda2 = 1/(5.25*4);  % Failure rate of function 2
    lambda3 = 1/(6.5*4);   % Failure rate of function 3

    % Penalty costs
    C_p = [0 -200 -500 -600 -300 -400 -400 -700];

    % Allocate memory for the matrices that will store the sensitivity
    % results before they are output to a file
    results = zeros(TotStates, 100);
    policies = cell(100);
```

```
for base = 1:100
    disp(base)
    % On orbit repair cost by state
    if parameter == 1
        Launch = base*10;
    else
        Launch = 450;
    end
    C_m = [-realmax -(Launch + 20) -(Launch + 15) -(Launch + 35) -(Launch + 35) -(Launch + 55)
           -(Launch + 50) -(Launch + 70)];

    % Satellite replacement cost
    if parameter == 2
        C_s = -base*10;
    else
        C_s = -500;
    end

    % Use when an action is not feasible for that state
    NA = -99999;
    % Failure probabilities for each function
    F = [1-exp(-lambda1) 1-exp(-lambda2) 1-exp(-lambda3)];
    % Survival probabilities for each function
    R = [exp(-lambda1) exp(-lambda2) exp(-lambda3)];
    % Probability of a successful repair for each function
    M = [.95 .96 .97];
    % Probability of a successful replacement for each function
    G = [.975 .94 .98];

    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    % Define Transition Probabilities
    % p(s,a,j) ==> Probability  of going to state j, given presently in
    %              state s and choosing action a
    p = zeros(TotStates, actions, TotStates);

    p(1,1,1) = R(1)*R(2)*R(3);
    p(1,1,2) = R(1)*R(2)*F(3);
    p(1,1,3) = R(1)*F(2)*R(3);
    p(1,1,4) = R(1)*F(2)*F(3);
    p(1,1,5) = F(1)*R(2)*R(3);
    p(1,1,6) = F(1)*R(2)*F(3);
    p(1,1,7) = F(1)*F(2)*R(3);
    p(1,1,8) = F(1)*F(2)*F(3);
```

```
% There is no need to include the transition probabilities for action 2
% while in state 2 because the action will never be chosen.  This is because
% the transition probabilities are identical to those of action 1, but the
% large setup cost is not incurred with action 1.

p(1,3,1) = G(1)*G(2)*G(3);
p(1,3,2) = G(1)*G(2)*(1-G(3));
p(1,3,3) = G(1)*(1-G(2))*G(3);
p(1,3,4) = G(1)*(1-G(2))*(1-G(3));
p(1,3,5) = (1-G(1))*G(2)*G(3);
p(1,3,6) = (1-G(1))*G(2)*(1-G(3));
p(1,3,7) = (1-G(1))*(1-G(2))*G(3);
p(1,3,8) = (1-G(1))*(1-G(2))*(1-G(3));

p(2,1,2) = R(1)*R(2);
p(2,1,4) = R(1)*F(2);
p(2,1,6) = F(1)*R(2);
p(2,1,8) = F(1)*F(2);

p(2,2,1) = R(1)*R(2)*M(3);
p(2,2,2) = R(1)*R(2)*(1-M(3));
p(2,2,3) = R(1)*F(2)*M(3);
p(2,2,4) = R(1)*F(2)*(1-M(3));
p(2,2,5) = F(1)*R(2)*M(3);
p(2,2,6) = F(1)*R(2)*(1-M(3));
p(2,2,7) = F(1)*F(2)*M(3);
p(2,2,8) = F(1)*F(2)*(1-M(3));

p(2,3,1) = G(1)*G(2)*G(3);
p(2,3,2) = G(1)*G(2)*(1-G(3));
p(2,3,3) = G(1)*(1-G(2))*G(3);
p(2,3,4) = G(1)*(1-G(2))*(1-G(3));
p(2,3,5) = (1-G(1))*G(2)*G(3);
p(2,3,6) = (1-G(1))*G(2)*(1-G(3));
p(2,3,7) = (1-G(1))*(1-G(2))*G(3);
p(2,3,8) = (1-G(1))*(1-G(2))*(1-G(3));

p(3,1,3) = R(1)*R(3);
p(3,1,4) = R(1)*F(3);
p(3,1,7) = F(1)*R(3);
p(3,1,8) = F(1)*F(3);
```

```
p(3,2,1) = R(1)*M(2)*R(3);
p(3,2,2) = R(1)*M(2)*F(3);
p(3,2,3) = R(1)*(1-M(2))*R(3);
p(3,2,4) = R(1)*(1-M(2))*F(3);
p(3,2,5) = F(1)*M(2)*R(3);
p(3,2,6) = F(1)*M(2)*F(3);
p(3,2,7) = F(1)*(1-M(2))*R(3);
p(3,2,8) = F(1)*(1-M(2))*F(3);

p(3,3,1) = G(1)*G(2)*G(3);
p(3,3,2) = G(1)*G(2)*(1-G(3));
p(3,3,3) = G(1)*(1-G(2))*G(3);
p(3,3,4) = G(1)*(1-G(2))*(1-G(3));
p(3,3,5) = (1-G(1))*G(2)*G(3);
p(3,3,6) = (1-G(1))*G(2)*(1-G(3));
p(3,3,7) = (1-G(1))*(1-G(2))*G(3);
p(3,3,8) = (1-G(1))*(1-G(2))*(1-G(3));

p(4,1,4) = R(1);
p(4,1,8) = F(1);

p(4,2,1) = R(1)*M(2)*M(3);
p(4,2,2) = R(1)*M(2)*(1-M(3));
p(4,2,3) = R(1)*(1-M(2))*M(3);
p(4,2,4) = R(1)*(1-M(2))*(1-M(3));
p(4,2,5) = F(1)*M(2)*M(3);
p(4,2,6) = F(1)*M(2)*(1-M(3));
p(4,2,7) = F(1)*(1-M(2))*M(3);
p(4,2,8) = F(1)*(1-M(2))*(1-M(3));

p(4,3,1) = G(1)*G(2)*G(3);
p(4,3,2) = G(1)*G(2)*(1-G(3));
p(4,3,3) = G(1)*(1-G(2))*G(3);
p(4,3,4) = G(1)*(1-G(2))*(1-G(3));
p(4,3,5) = (1-G(1))*G(2)*G(3);
p(4,3,6) = (1-G(1))*G(2)*(1-G(3));
p(4,3,7) = (1-G(1))*(1-G(2))*G(3);
p(4,3,8) = (1-G(1))*(1-G(2))*(1-G(3));

p(5,1,5) = R(2)*R(3);
p(5,1,6) = R(2)*F(3);
p(5,1,7) = F(2)*R(3);
p(5,1,8) = F(2)*F(3);
```

```
p(5,2,1) = M(1)*R(2)*R(3);
p(5,2,2) = M(1)*R(2)*F(3);
p(5,2,3) = M(1)*F(2)*R(3);
p(5,2,4) = M(1)*F(2)*F(3);
p(5,2,5) = (1-M(1))*R(2)*R(3);
p(5,2,6) = (1-M(1))*R(2)*F(3);
p(5,2,7) = (1-M(1))*F(2)*R(3);
p(5,2,8) = (1-M(1))*F(2)*F(3);

p(5,3,1) = G(1)*G(2)*G(3);
p(5,3,2) = G(1)*G(2)*(1-G(3));
p(5,3,3) = G(1)*(1-G(2))*G(3);
p(5,3,4) = G(1)*(1-G(2))*(1-G(3));
p(5,3,5) = (1-G(1))*G(2)*G(3);
p(5,3,6) = (1-G(1))*G(2)*(1-G(3));
p(5,3,7) = (1-G(1))*(1-G(2))*G(3);
p(5,3,8) = (1-G(1))*(1-G(2))*(1-G(3));

p(6,1,6) = R(2);
p(6,1,8) = F(2);

p(6,2,1) = M(1)*R(2)*M(3);
p(6,2,2) = M(1)*R(2)*(1-M(3));
p(6,2,3) = M(1)*F(2)*M(3);
p(6,2,4) = M(1)*F(2)*(1-M(3));
p(6,2,5) = (1-M(1))*R(2)*M(3);
p(6,2,6) = (1-M(1))*R(2)*(1-M(3));
p(6,2,7) = (1-M(1))*F(2)*M(3);
p(6,2,8) = (1-M(1))*F(2)*(1-M(3));

p(6,3,1) = G(1)*G(2)*G(3);
p(6,3,2) = G(1)*G(2)*(1-G(3));
p(6,3,3) = G(1)*(1-G(2))*G(3);
p(6,3,4) = G(1)*(1-G(2))*(1-G(3));
p(6,3,5) = (1-G(1))*G(2)*G(3);
p(6,3,6) = (1-G(1))*G(2)*(1-G(3));
p(6,3,7) = (1-G(1))*(1-G(2))*G(3);
p(6,3,8) = (1-G(1))*(1-G(2))*(1-G(3));

p(7,1,7) = R(3);
p(7,1,8) = F(3);
```

```
p(7,2,1) = M(1)*M(2)*R(3);
p(7,2,2) = M(1)*M(2)*F(3);
p(7,2,3) = M(1)*(1-M(2))*R(3);
p(7,2,4) = M(1)*(1-M(2))*F(3);
p(7,2,5) = (1-M(1))*M(2)*R(3);
p(7,2,6) = (1-M(1))*M(2)*F(3);
p(7,2,7) = (1-M(1))*(1-M(2))*R(3);
p(7,2,8) = (1-M(1))*(1-M(2))*F(3);

p(7,3,1) = G(1)*G(2)*G(3);
p(7,3,2) = G(1)*G(2)*(1-G(3));
p(7,3,3) = G(1)*(1-G(2))*G(3);
p(7,3,4) = G(1)*(1-G(2))*(1-G(3));
p(7,3,5) = (1-G(1))*G(2)*G(3);
p(7,3,6) = (1-G(1))*G(2)*(1-G(3));
p(7,3,7) = (1-G(1))*(1-G(2))*G(3);
p(7,3,8) = (1-G(1))*(1-G(2))*(1-G(3));

p(8,1,8) = 1;

p(8,2,1) = M(1)*M(2)*M(3);
p(8,2,2) = M(1)*M(2)*(1-M(3));
p(8,2,3) = M(1)*(1-M(2))*M(3);
p(8,2,4) = M(1)*(1-M(2))*(1-M(3));
p(8,2,5) = (1-M(1))*M(2)*M(3);
p(8,2,6) = (1-M(1))*M(2)*(1-M(3));
p(8,2,7) = (1-M(1))*(1-M(2))*M(3);
p(8,2,8) = (1-M(1))*(1-M(2))*(1-M(3));

p(8,3,1) = G(1)*G(2)*G(3);
p(8,3,2) = G(1)*G(2)*(1-G(3));
p(8,3,3) = G(1)*(1-G(2))*G(3);
p(8,3,4) = G(1)*(1-G(2))*(1-G(3));
p(8,3,5) = (1-G(1))*G(2)*G(3);
p(8,3,6) = (1-G(1))*G(2)*(1-G(3));
p(8,3,7) = (1-G(1))*(1-G(2))*G(3);
p(8,3,8) = (1-G(1))*(1-G(2))*(1-G(3));

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Define the Immediate Rewards - deterministic part of the reward
% function ra{a}(s) ==> Immediate reward when choosing action a in
% state s
```

```
ra = cell(actions);

ra{1} = zeros(TotStates);
ra{2} = zeros(TotStates);
ra{3} = zeros(TotStates);

ra{2}(1) = NA;
ra{2}(2) = C_m(2);
ra{2}(3) = C_m(3);
ra{2}(4) = C_m(4);
ra{2}(5) = C_m(5);
ra{2}(6) = C_m(6);
ra{2}(7) = C_m(7);
ra{2}(8) = C_m(8);

ra{3}(1) = C_s;
ra{3}(2) = C_s;
ra{3}(3) = C_s;
ra{3}(4) = C_s;
ra{3}(5) = C_s;
ra{3}(6) = C_s;
ra{3}(7) = C_s;
ra{3}(8) = C_s;

% Define the Expected Rewards
% r(s,a) ==> The expected reward when in state s and choosing
% action a. This is equivalent to the immediate reward plus the
% expected penalty cost over the next period.
r = zeros(TotStates,actions);

for s = 1:TotStates
    for a = 1:actions

        % Calculate the expected penalty cost over the next period
        % (the sum of the penalty costs times the probability that
        % that cost is incurred).
        Expected_Penalty = 0;

        for j = 1:TotStates
            Expected_Penalty = Expected_Penalty + p(s,a,j)*C_p(j);
        end

        % Assign the expected reward resulting from taking action a
```

```
        % in state s.
        if Expected_Penalty == 0
            r(s,a) = NA; % The action is not feasible
        else
            r(s,a) = ra{a}(s) + Expected_Penalty;
        end
    end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Backward Induction Code: Directly adapted from Capt Sumter's code

% Define the u, ustar and dstar vector
u = zeros(TotStates,actions,N);
ustar = zeros(TotStates,N);
dstar = zeros(TotStates,N);
% Note: The value of any action in the final time period has zero
% reward and therefore zero utility
% ustar(s,N) = 0 for all s

tic % Starts the timer used to measure the time to solve the MDP

n = N - 1;
while n >= 1
    for s = 1:TotStates
        ustar(s,n) = -realmax;
        for a = 1:actions
            if r(s,a) ~= NA  % Infeasible action - no reward
                expected_value = 0;
                for j = 1:TotStates
                    expected_value = expected_value + p(s,a,j)*ustar(j,n+1);
                end
                u(s,a,n) = r(s,a) + expected_value;

                if u(s,a,n) > ustar(s,n)
                    ustar(s,n) = u(s,a,n);
                    dstar(s,n) = a;
                end
            end
        end
    end
    n = n - 1;   % Decrement the time
end
```

```
    toc % Stops the timer used to measure the time to solve the MDP

    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    % Store the optimal value (ustar) results in the matrix 'results'
    results(1, base) = base;
    for index = 1:TotStates
        results(index, base) = -ustar(index,1);
    end

    % Store the optimal policies (dstar) in the cell array "policies"
    policies{base} = dstar;
end % end the loop that controls the sensitivity analysis

% Determine which file to write to
if parameter == 1
    fid = fopen('One_Sat_Repair_Cost.txt','w');
else
    fid = fopen('One_Sat_Replace_Cost.txt','w');
end

% Write the results to a file in the current directory
for j=1:base
    fprintf(fid, '%8.5f\t', j);
end
fprintf(fid, '\n');
for i = 1:TotStates;
    for j=1:base
        fprintf(fid, '%8.5f\t',results(i,j));
    end
    fprintf(fid, '\n');
end

for j=1:base
    fprintf(fid, '\n Parameter Iteration - ');
    fprintf(fid, '%g\t', j);
    fprintf(fid, '\n');
    for s=1:TotStates
        for n=1:N
            fprintf(fid, '%g\t', policies{j}(s,n));
        end
        fprintf(fid, '\n');
    end
```

```
    end
    % Close file
    fclose(fid);
end
```

Appendix D. Three-Satellite One-Way Analysis

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Author: 1Lt Tim Cook
%         AFIT/ENS/GOR-05M
%         March 2005
% This program performs a one-way sensitivity analysis on a Markov
% decision process formulated to find an optimal maintenance policy for a
% three satellite constellation.  In this file, the parameters that are
% modified are the space vehicle cost and the replacement cost.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Define Parameters
N = 20;     % Length of planning horizon
M = [2 2 3];% Number of functions desired operational in each satellite
K = 3;      % Number of Satellites

% Create a vector that keeps the number of degradation levels for each
% satellite
states = zeros(K); for i=1:K
    states(i) = 2^M(i); % Number of possible states for satellite i
end

% Define the total number of states that the entire satellite
% constellation can possibly enter into
TotStates = prod(states(:,1));

% Number of possible actions for each satellite
actions = 3;

% Failure rates -- 1/(expected lifetime in years)*(4 quarters)
lambda1 = 1/(5.5*4);   % Failure rate of function 1
lambda2 = 1/(5.25*4);  % Failure rate of function 2
lambda3 = 1/(6.5*4);   % Failure rate of function 3

% Allocate memory for the matrices that will store the sensitivity
% results before they are output to a file
results = zeros(TotStates, 100); policies = cell(100);

% Use when an action is not feasible for that state
NA = -99999;
```

```
% Failure probabilities for each function
F = [1-exp(-lambda1) 1-exp(-lambda2) 1-exp(-lambda3)];
% Survival probabilities for each function
R = [exp(-lambda1) exp(-lambda2) exp(-lambda3)];
% Probability of a successful repair for each function
M = [.95 .96 .97];
% Probability of a successful replacement for each function
G = [.975 .94 .98];

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Define Transition Probabilities
% p{k}(s,a,j) ==> Probability  for satellite k of going to state j,
%                 given presently in state s and choosing action a for
%                 that individual satellite

% Allocate memory for the transition probability matricies
% using cell arrays
p = cell(K); for i = 1:K
    p(i) = {zeros(states(i), actions, states(i))};
end

% Define the transition probabilities
p{1}(1,1,1) = R(1)*R(2);
p{1}(1,1,2) = R(1)*F(2);
p{1}(1,1,3) = F(1)*R(2);
p{1}(1,1,4) = F(1)*F(2);

% There is no need to include the transition probabilities for action 2
% while in state 2 because the action will never be chosen.  This is because
% the transition probabilities are identical to those of action 1, but the
% large setup cost is not incurred with action 1.  The same holds for the other
% two satellites.

p{1}(1,3,1) = G(1)*G(2);
p{1}(1,3,2) = G(1)*(1-G(2));
p{1}(1,3,3) = (1-G(1))*G(2);
p{1}(1,3,4) = (1-G(1))*(1-G(2));

p{1}(2,1,2) = R(1);
p{1}(2,1,4) = F(1);

p{1}(2,2,1) = R(1)*M(2);
p{1}(2,2,2) = R(1)*(1-M(2));
```

```
p{1}(2,2,3) = F(1)*M(2);
p{1}(2,2,4) = F(1)*(1-M(2));

p{1}(2,3,1) = G(1)*G(2);
p{1}(2,3,2) = G(1)*(1-G(2));
p{1}(2,3,3) = (1-G(1))*G(2);
p{1}(2,3,4) = (1-G(1))*(1-G(2));

p{1}(3,1,3) = R(2);
p{1}(3,1,4) = F(2);

p{1}(3,2,1) = M(1)*R(2);
p{1}(3,2,2) = M(1)*F(2);
p{1}(3,2,3) = (1-M(1))*R(2);
p{1}(3,2,4) = (1-M(1))*F(2);

p{1}(3,3,1) = G(1)*G(2);
p{1}(3,3,2) = G(1)*(1-G(2));
p{1}(3,3,3) = (1-G(1))*G(2);
p{1}(3,3,4) = (1-G(1))*(1-G(2));

p{1}(4,1,4) = 1;

p{1}(4,2,1) = M(1)*M(2);
p{1}(4,2,2) = M(1)*(1-M(2));
p{1}(4,2,3) = (1-M(1))*M(2);
p{1}(4,2,4) = (1-M(1))*(1-M(2));

p{1}(4,3,1) = G(1)*G(2);
p{1}(4,3,2) = G(1)*(1-G(2));
p{1}(4,3,3) = (1-G(1))*G(2);
p{1}(4,3,4) = (1-G(1))*(1-G(2));

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

p{2}(1,1,1) = R(1)*R(2);
p{2}(1,1,2) = R(1)*F(2);
p{2}(1,1,3) = F(1)*R(2);
p{2}(1,1,4) = F(1)*F(2);

p{2}(1,3,1) = G(1)*G(2);
p{2}(1,3,2) = G(1)*(1-G(2));
p{2}(1,3,3) = (1-G(1))*G(2);
```

```
p{2}(1,3,4) = (1-G(1))*(1-G(2));

p{2}(2,1,2) = R(1);
p{2}(2,1,4) = F(1);

p{2}(2,2,1) = R(1)*M(2);
p{2}(2,2,2) = R(1)*(1-M(2));
p{2}(2,2,3) = F(1)*M(2);
p{2}(2,2,4) = F(1)*(1-M(2));

p{2}(2,3,1) = G(1)*G(2);
p{2}(2,3,2) = G(1)*(1-G(2));
p{2}(2,3,3) = (1-G(1))*G(2);
p{2}(2,3,4) = (1-G(1))*(1-G(2));

p{2}(3,1,3) = R(2);
p{2}(3,1,4) = F(2);

p{2}(3,2,1) = M(1)*R(2);
p{2}(3,2,2) = M(1)*F(2);
p{2}(3,2,3) = (1-M(1))*R(2);
p{2}(3,2,4) = (1-M(1))*F(2);

p{2}(3,3,1) = G(1)*G(2);
p{2}(3,3,2) = G(1)*(1-G(2));
p{2}(3,3,3) = (1-G(1))*G(2);
p{2}(3,3,4) = (1-G(1))*(1-G(2));

p{2}(4,1,4) = 1;

p{2}(4,2,1) = M(1)*M(2);
p{2}(4,2,2) = M(1)*(1-M(2));
p{2}(4,2,3) = (1-M(1))*M(2);
p{2}(4,2,4) = (1-M(1))*(1-M(2));

p{2}(4,3,1) = G(1)*G(2);
p{2}(4,3,2) = G(1)*(1-G(2));
p{2}(4,3,3) = (1-G(1))*G(2);
p{2}(4,3,4) = (1-G(1))*(1-G(2));

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

p{3}(1,1,1) = R(1)*R(2)*R(3); p{3}(1,1,2) = R(1)*R(2)*F(3);
```

```
p{3}(1,1,3) = R(1)*F(2)*R(3); p{3}(1,1,4) = R(1)*F(2)*F(3);
p{3}(1,1,5) = F(1)*R(2)*R(3); p{3}(1,1,6) = F(1)*R(2)*F(3);
p{3}(1,1,7) = F(1)*F(2)*R(3); p{3}(1,1,8) = F(1)*F(2)*F(3);

p{3}(1,3,1) = G(1)*G(2)*G(3);
p{3}(1,3,2) = G(1)*G(2)*(1-G(3));
p{3}(1,3,3) = G(1)*(1-G(2))*G(3);
p{3}(1,3,4) = G(1)*(1-G(2))*(1-G(3));
p{3}(1,3,5) = (1-G(1))*G(2)*G(3);
p{3}(1,3,6) = (1-G(1))*G(2)*(1-G(3));
p{3}(1,3,7) = (1-G(1))*(1-G(2))*G(3);
p{3}(1,3,8) = (1-G(1))*(1-G(2))*(1-G(3));

p{3}(2,1,2) = R(1)*R(2);
p{3}(2,1,4) = R(1)*F(2);
p{3}(2,1,6) = F(1)*R(2);
p{3}(2,1,8) = F(1)*F(2);

p{3}(2,2,1) = R(1)*R(2)*M(3);
p{3}(2,2,2) = R(1)*R(2)*(1-M(3));
p{3}(2,2,3) = R(1)*F(2)*M(3);
p{3}(2,2,4) = R(1)*F(2)*(1-M(3));
p{3}(2,2,5) = F(1)*R(2)*M(3);
p{3}(2,2,6) = F(1)*R(2)*(1-M(3));
p{3}(2,2,7) = F(1)*F(2)*M(3);
p{3}(2,2,8) = F(1)*F(2)*(1-M(3));

p{3}(2,3,1) = G(1)*G(2)*G(3);
p{3}(2,3,2) = G(1)*G(2)*(1-G(3));
p{3}(2,3,3) = G(1)*(1-G(2))*G(3);
p{3}(2,3,4) = G(1)*(1-G(2))*(1-G(3));
p{3}(2,3,5) = (1-G(1))*G(2)*G(3);
p{3}(2,3,6) = (1-G(1))*G(2)*(1-G(3));
p{3}(2,3,7) = (1-G(1))*(1-G(2))*G(3);
p{3}(2,3,8) = (1-G(1))*(1-G(2))*(1-G(3));

p{3}(3,1,3) = R(1)*R(3);
p{3}(3,1,4) = R(1)*F(3);
p{3}(3,1,7) = F(1)*R(3);
p{3}(3,1,8) = F(1)*F(3);

p{3}(3,2,1) = R(1)*M(2)*R(3);
p{3}(3,2,2) = R(1)*M(2)*F(3);
```

```
p{3}(3,2,3) = R(1)*(1-M(2))*R(3);
p{3}(3,2,4) = R(1)*(1-M(2))*F(3);
p{3}(3,2,5) = F(1)*M(2)*R(3);
p{3}(3,2,6) = F(1)*M(2)*F(3);
p{3}(3,2,7) = F(1)*(1-M(2))*R(3);
p{3}(3,2,8) = F(1)*(1-M(2))*F(3);

p{3}(3,3,1) = G(1)*G(2)*G(3);
p{3}(3,3,2) = G(1)*G(2)*(1-G(3));
p{3}(3,3,3) = G(1)*(1-G(2))*G(3);
p{3}(3,3,4) = G(1)*(1-G(2))*(1-G(3));
p{3}(3,3,5) = (1-G(1))*G(2)*G(3);
p{3}(3,3,6) = (1-G(1))*G(2)*(1-G(3));
p{3}(3,3,7) = (1-G(1))*(1-G(2))*G(3);
p{3}(3,3,8) = (1-G(1))*(1-G(2))*(1-G(3));

p{3}(4,1,4) = R(1);
p{3}(4,1,8) = F(1);

p{3}(4,2,1) = R(1)*M(2)*M(3);
p{3}(4,2,2) = R(1)*M(2)*(1-M(3));
p{3}(4,2,3) = R(1)*(1-M(2))*M(3);
p{3}(4,2,4) = R(1)*(1-M(2))*(1-M(3));
p{3}(4,2,5) = F(1)*M(2)*M(3);
p{3}(4,2,6) = F(1)*M(2)*(1-M(3));
p{3}(4,2,7) = F(1)*(1-M(2))*M(3);
p{3}(4,2,8) = F(1)*(1-M(2))*(1-M(3));

p{3}(4,3,1) = G(1)*G(2)*G(3);
p{3}(4,3,2) = G(1)*G(2)*(1-G(3));
p{3}(4,3,3) = G(1)*(1-G(2))*G(3);
p{3}(4,3,4) = G(1)*(1-G(2))*(1-G(3));
p{3}(4,3,5) = (1-G(1))*G(2)*G(3);
p{3}(4,3,6) = (1-G(1))*G(2)*(1-G(3));
p{3}(4,3,7) = (1-G(1))*(1-G(2))*G(3);
p{3}(4,3,8) = (1-G(1))*(1-G(2))*(1-G(3));

p{3}(5,1,5) = R(2)*R(3);
p{3}(5,1,6) = R(2)*F(3);
p{3}(5,1,7) = F(2)*R(3);
p{3}(5,1,8) = F(2)*F(3);

p{3}(5,2,1) = M(1)*R(2)*R(3);
```

```
p{3}(5,2,2) = M(1)*R(2)*F(3);
p{3}(5,2,3) = M(1)*F(2)*R(3);
p{3}(5,2,4) = M(1)*F(2)*F(3);
p{3}(5,2,5) = (1-M(1))*R(2)*R(3);
p{3}(5,2,6) = (1-M(1))*R(2)*F(3);
p{3}(5,2,7) = (1-M(1))*F(2)*R(3);
p{3}(5,2,8) = (1-M(1))*F(2)*F(3);

p{3}(5,3,1) = G(1)*G(2)*G(3);
p{3}(5,3,2) = G(1)*G(2)*(1-G(3));
p{3}(5,3,3) = G(1)*(1-G(2))*G(3);
p{3}(5,3,4) = G(1)*(1-G(2))*(1-G(3));
p{3}(5,3,5) = (1-G(1))*G(2)*G(3);
p{3}(5,3,6) = (1-G(1))*G(2)*(1-G(3));
p{3}(5,3,7) = (1-G(1))*(1-G(2))*G(3);
p{3}(5,3,8) = (1-G(1))*(1-G(2))*(1-G(3));

p{3}(6,1,6) = R(2);
p{3}(6,1,8) = F(2);

p{3}(6,2,1) = M(1)*R(2)*M(3);
p{3}(6,2,2) = M(1)*R(2)*(1-M(3));
p{3}(6,2,3) = M(1)*F(2)*M(3);
p{3}(6,2,4) = M(1)*F(2)*(1-M(3));
p{3}(6,2,5) = (1-M(1))*R(2)*M(3);
p{3}(6,2,6) = (1-M(1))*R(2)*(1-M(3));
p{3}(6,2,7) = (1-M(1))*F(2)*M(3);
p{3}(6,2,8) = (1-M(1))*F(2)*(1-M(3));

p{3}(6,3,1) = G(1)*G(2)*G(3);
p{3}(6,3,2) = G(1)*G(2)*(1-G(3));
p{3}(6,3,3) = G(1)*(1-G(2))*G(3);
p{3}(6,3,4) = G(1)*(1-G(2))*(1-G(3));
p{3}(6,3,5) = (1-G(1))*G(2)*G(3);
p{3}(6,3,6) = (1-G(1))*G(2)*(1-G(3));
p{3}(6,3,7) = (1-G(1))*(1-G(2))*G(3);
p{3}(6,3,8) = (1-G(1))*(1-G(2))*(1-G(3));

p{3}(7,1,7) = R(3);
p{3}(7,1,8) = F(3);

p{3}(7,2,1) = M(1)*M(2)*R(3);
p{3}(7,2,2) = M(1)*M(2)*F(3);
```

```
p{3}(7,2,3) = M(1)*(1-M(2))*R(3);
p{3}(7,2,4) = M(1)*(1-M(2))*F(3);
p{3}(7,2,5) = (1-M(1))*M(2)*R(3);
p{3}(7,2,6) = (1-M(1))*M(2)*F(3);
p{3}(7,2,7) = (1-M(1))*(1-M(2))*R(3);
p{3}(7,2,8) = (1-M(1))*(1-M(2))*F(3);

p{3}(7,3,1) = G(1)*G(2)*G(3);
p{3}(7,3,2) = G(1)*G(2)*(1-G(3));
p{3}(7,3,3) = G(1)*(1-G(2))*G(3);
p{3}(7,3,4) = G(1)*(1-G(2))*(1-G(3));
p{3}(7,3,5) = (1-G(1))*G(2)*G(3);
p{3}(7,3,6) = (1-G(1))*G(2)*(1-G(3));
p{3}(7,3,7) = (1-G(1))*(1-G(2))*G(3);
p{3}(7,3,8) = (1-G(1))*(1-G(2))*(1-G(3));

p{3}(8,1,8) = 1;

p{3}(8,2,1) = M(1)*M(2)*M(3);
p{3}(8,2,2) = M(1)*M(2)*(1-M(3));
p{3}(8,2,3) = M(1)*(1-M(2))*M(3);
p{3}(8,2,4) = M(1)*(1-M(2))*(1-M(3));
p{3}(8,2,5) = (1-M(1))*M(2)*M(3);
p{3}(8,2,6) = (1-M(1))*M(2)*(1-M(3));
p{3}(8,2,7) = (1-M(1))*(1-M(2))*M(3);
p{3}(8,2,8) = (1-M(1))*(1-M(2))*(1-M(3));

p{3}(8,3,1) = G(1)*G(2)*G(3);
p{3}(8,3,2) = G(1)*G(2)*(1-G(3));
p{3}(8,3,3) = G(1)*(1-G(2))*G(3);
p{3}(8,3,4) = G(1)*(1-G(2))*(1-G(3));
p{3}(8,3,5) = (1-G(1))*G(2)*G(3);
p{3}(8,3,6) = (1-G(1))*G(2)*(1-G(3));
p{3}(8,3,7) = (1-G(1))*(1-G(2))*G(3);
p{3}(8,3,8) = (1-G(1))*(1-G(2))*(1-G(3));

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Define the state space for each satellite
Sat_s = cell(K); Sat_s{1} = [1 1
    1 0
    0 1
    0 0];
Sat_s{2} = [1 1
```

```
    1 0
    0 1
    0 0];
Sat_s{3} = [1 1 1
    1 1 0
    1 0 1
    1 0 0
    0 1 1
    0 1 0
    0 0 1
    0 0 0];

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Define the overall state space for the constellation
Constellation_s = cell(TotStates);
s1 = 1;
s2 = 1;
s3 = 1;
for i = 1:TotStates
    Constellation_s{i} = [s1 s2 s3];

    s3 = s3 + 1;
    if s3 > states(3)
        s3 = 1;
        s2 = s2 + 1;
    end
    if s2 > states(2)
        s2 = 1;
        s1 = s1 + 1;
    end
    if s1 > states(1)
        s1 = 1;
    end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Penalty Cost assigned per state
% The decision maker desires that
C_p = zeros(TotStates); for i = 1:TotStates

    % Function 1 Penalty Cost
    s1 = Constellation_s{i}(1);
    s2 = Constellation_s{i}(2);
```

```
    s3 = Constellation_s{i}(3);
    sum = Sat_s{1}(s1,1)+Sat_s{2}(s2,1)+Sat_s{3}(s3,1);
    if sum == 3
        C_p(i) = C_p(i) + 0;
    elseif sum == 2
        C_p(i) = C_p(i) + 0;
    elseif sum == 1
        C_p(i) = C_p(i) - 500;
    elseif sum == 0
        C_p(i) = C_p(i) - 650;
    end

    % Function 2 Penalty Cost
    sum = Sat_s{1}(s1,2)+Sat_s{2}(s2,2)+Sat_s{3}(s3,2);
    if sum == 3
        C_p(i) = C_p(i) + 0;
    elseif sum == 2
        C_p(i) = C_p(i) + 0;
    elseif sum == 1
        C_p(i) = C_p(i) - 450;
    elseif sum == 0
        C_p(i) = C_p(i) - 700;
    end

    % Function 3 Penalty Cost
    if Sat_s{3}(s3,3) == 1
        C_p(i) = C_p(i) + 0;
    elseif Sat_s{3}(s3,3) == 0
        C_p(i) = C_p(i) - 200;
    end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Define the overall constellation action set in a cell array
Constellation_a = cell(actions^K); a1=1; a2=1; a3=1; for i =
1:actions^K
    Constellation_a{i}=[a1 a2 a3];

    a3 = a3 + 1;
    if a3 > actions
        a2 = a2 + 1;
        a3 = 1;
```

```
    end
    if a2 > actions
        a1 = a1 + 1;
        a2 = 1;
    end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Define the overall constellation transition probabilities.  Do this
% by looping through each possibile (s,a,j) combination in the
% constellation and multiplying the individual satellite transition
% probabilities that would lead to the overall satellite constellation
% transition.  The user of this code must know clearly what state the
% index of each satellite's state and action vector represents.
s1 = 1; s2 = 1; s3 = 1; a1 = 1; a2 = 1; a3 = 1; j1 = 1; j2 = 1; j3
= 1; Constellation_p = zeros(TotStates,actions^K,TotStates);

for s = 1:TotStates
    for a = 1:actions^K
        for j = 1:TotStates

            Constellation_p(s,a,j) = p{1}(s1,a1,j1)*p{2}(s2,a2,j2)*p{3}(s3,a3,j3);

            % Eliminate transition probabilities associated with
            % infeasible actions. Set the transition probabilities to
            % 0 for all infeasible actions and therefore sets the
            % expected rewards to 0 so they can be identified and then
            % set to NA.
            if s1 == 1
                if a1 == 2
                    Constellation_p(s,a,j) = 0;
                end
            end
            if s2 == 1
                if a2 == 2
                    Constellation_p(s,a,j) = 0;
                end
            end
            if s3 == 1
                if a3 == 2
                    Constellation_p(s,a,j) = 0;
                end
            end
```

```
% The previous block eliminates any on orbit repairs to a
% satellite that has nothing broken.

if s1 == states(1)
    if s2 == states(2)
        if s3 == states(3)
            if a1 == 1
                if a2 == 1
                    if a3 == 1
                        Constellation_p(s,a,j) = 0;
                    end
                end
            end
        end
    end
end
% The previous block eliminates the possibility of doing
% nothing when all functions of all satellites are
% non-operational.

j3 = j3 + 1;
if j3 > states(3)
    j3 = 1;
    j2 = j2 + 1;
end
if j2 > states(2)
    j2 = 1;
    j1 = j1 + 1;
end
if j1 > states(1)
    j1 = 1;
    a3 = a3 + 1;
end
if a3 > actions
    a3 = 1;
    a2 = a2 + 1;
end
if a2 > actions
    a2 = 1;
    a1 = a1 + 1;
end
if a1 > actions
    a1 = 1;
```

```
            s3 = s3 + 1;
        end
        if s3 > states(3)
            s3 = 1;
            s2 = s2 + 1;
        end
        if s2 > states(2)
            s2 = 1;
            s1 = s1 + 1;
        end
        if s1 > states(1)
            s1 = 1;
        end
    end
  end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Sensitivity analysis portion of the code
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Begin the big for loop that performs the sensitivity analysis
for parameter = 1:2
    for base = 1:100
        disp(base)
        % On orbit repair cost by state
        if parameter == 1
            SpaceVehicle = -base*10;
        else
            SpaceVehicle = -450;
        end
        C_m = cell(K);
        C_m{1} = [-realmax SpaceVehicle-15 SpaceVehicle-35 SpaceVehicle-50];
        C_m{2} = [-realmax SpaceVehicle-15 SpaceVehicle-35 SpaceVehicle-50];
        C_m{3} = [-realmax SpaceVehicle-20 SpaceVehicle-15 SpaceVehicle-35 SpaceVehicle-35 SpaceVehicle-55
                 SpaceVehicle-50 SpaceVehicle-70];

        % Satellite replacement cost
        if parameter == 2
            C_s = [-base*10 -base*10 (-base*10)-50];
        else
            C_s = [-500 -500 -520];
        end
```

```
disp(C_s)
disp(C_m{1}(2))
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Define the Immediate Rewards for each individual satellite
% ra(a,s) ==> Immediate reward when choosing action a while in
% state s (constellation).
% Assume that the repair costs are the same for either function on
% all satellites
ra = zeros(actions^K, TotStates);

for Cons_A = 1:actions^K
    for Cons_S = 1:TotStates
        for k = 1:K

            if Constellation_a{Cons_A}(k) == 1
                ra(Cons_A,Cons_S) = ra(Cons_A,Cons_S) + 0;
            end
            if Constellation_a{Cons_A}(k) == 2
                SatState = Constellation_s{Cons_S}(k);
                ra(Cons_A,Cons_S) = ra(Cons_A,Cons_S) + C_m{k}(SatState);
            end
            if Constellation_a{Cons_A}(k) == 3
                ra(Cons_A,Cons_S) = ra(Cons_A,Cons_S) + C_s(k);
            end
        end
    end
end


% Define the Expected Rewards
% r(s,a) ==> The expected reward when in state s and choosing
% action a for the constellation. This is equivalent to the
% immediate reward plus the expected penalty cost over the next
% period.
r = zeros(TotStates,actions^K);

for s = 1:TotStates
    for a = 1:actions^K

        % Calculate the expected penalty cost over the next period
        % (the sum of the penalty costs times the probability that
```

```
        % that cost is incurred).
        Expected_penalty = 0;

        for j = 1:TotStates
            Expected_penalty = Expected_penalty + Constellation_p(s,a,j)*C_p(j);
        end

        % Assign the expected reward resulting from taking action a
        % in state s.
        if Expected_penalty == 0
            r(s,a) = NA;
        else
            r(s,a) = ra(a,s) + Expected_penalty;
        end
    end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Backward Induction Code

% Define the u, ustar and dstar vector
u = zeros(TotStates,actions,N);
ustar = zeros(TotStates,N);
dstar = zeros(TotStates,N);
% Note: The value of any action in the final time period has zero
% reward and therefore zero utility
% ustar(s,N) = 0 for all s

n = N - 1;
while n >= 1
    for s = 1:TotStates  % Loop through all constellation states
        ustar(s,n) = -realmax;
        for a = 1:actions^K  % Loop through all actions
            if r(s,a) ~= NA  % Infeasible action - no reward

                expected_value = 0;
                for j = 1:TotStates
                    expected_value = expected_value + Constellation_p(s,a,j)*ustar(j,n+1);
                end % End For Loop for expected value
                u(s,a,n) = r(s,a) + expected_value;

                if u(s,a,n) > ustar(s,n)
                    ustar(s,n) = u(s,a,n);
```

```
                    dstar(s,n) = a;
                end
            end
        end
    end
    n = n - 1;
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Store the optimal value results in the matrix 'results'
results(1, base) = base;
for index = 1:TotStates
    results(index, base) = ustar(index,1);
end
% Store the optimal policies in the cell array "policies"
policies{base} = dstar;

clear ustar;
clear dstar;

end % end the loop that controls the sensitivity analysis

% Determine which file to write to
if parameter == 1
    fid = fopen('Three_Sat_Repair_Cost.txt','w');
else
    fid = fopen('Three_Sat_Replace_Cost.txt','w');
end

% Write the results
for j=1:base
    fprintf(fid, '%8.5f\t', j);
end
fprintf(fid, '\n');
for i = 1:TotStates;
    for j=1:base
        fprintf(fid, '%8.5f\t',results(i,j));
    end
    fprintf(fid, '\n');
end

for j=1:base
    fprintf(fid, '\n Parameter Iteration - ');
```

```
        fprintf(fid, '%g\t', j);
        fprintf(fid, '\n');
        for s=1:TotStates
            for n=1:N
                fprintf(fid, '%g\t', policies{j}(s,n));
            end
            fprintf(fid, '\n');
        end
    end
    % Close file
    fclose(fid);
end
```

Appendix E. One-Satellite Two-Way Analysis

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Author: 1Lt Tim Cook
%         AFIT/ENS/GOR-05M
%         March 2005
% This program performs a two-way sensitivity analysis on a Markov decision
% process formulated to find an optimal maintenance policy for a one
% satellite constellation.  In this file, the parameters that are modified
% are the repair costs and the replacement costs.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Define Parameters
N = 20;  % Length of planning horizon
M = 3;   % Number of functions

% Total number of states that the satellite can possibly enter into
TotStates = 2^M;

% Number of possible actions
actions = 3;

% Failure rates -- 1/(expected lifetime in years)*(4 quarters)
lambda1 = 1/(5.5*4);   % Failure rate of function 1
lambda2 = 1/(5.25*4);  % Failure rate of function 2
lambda3 = 1/(6.5*4);   % Failure rate of function 3

% Penalty costs
C_p = [0 -200 -500 -600 -300 -400 -400 -700];

% Allocate memory for the matrices that will store the sensitivity
% results before they are output to a file
results = zeros(20, 100); policies = cell(2000);

% Begin the outer for loop that varies the satellite replacement cost.
for C_L = 1:20

    % Satellite replacement cost
    C_s = -50*C_L;

    % Begin the inner for loop that varies the space vehicle cost from 0%
    % to 100% of the satellite replacement cost.
    for C_V = 1:101
```

```
disp([C_L,C_V])

% On-orbit repair costs
Launch = -((101-C_V)/100)*C_s;
C_m = [-realmax -(Launch + 20) -(Launch + 15) -(Launch + 35) -(Launch + 35) -(Launch + 55)
       -(Launch + 50) -(Launch + 70)];
disp([C_s,Launch])

% Use when an action is not feasible for that state
NA = -99999;

% Failure probabilities for each function
F = [1-exp(-lambda1) 1-exp(-lambda2) 1-exp(-lambda3)];
% Survival probabilities for each function
R = [exp(-lambda1) exp(-lambda2) exp(-lambda3)];
% Probability of a successful repair for each function
M = [.95 .96 .97];
% Probability of a successful replacement for each function
G = [.975 .94 .98];

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Define Transition Probabilities
% p(s,a,j) ==> Probability  of going to state j, given presently in
%              state s and choosing action a
p = zeros(TotStates, actions, TotStates);

p(1,1,1) = R(1)*R(2)*R(3);
p(1,1,2) = R(1)*R(2)*F(3);
p(1,1,3) = R(1)*F(2)*R(3);
p(1,1,4) = R(1)*F(2)*F(3);
p(1,1,5) = F(1)*R(2)*R(3);
p(1,1,6) = F(1)*R(2)*F(3);
p(1,1,7) = F(1)*F(2)*R(3);
p(1,1,8) = F(1)*F(2)*F(3);

% There is no need to include the transition probabilities for action 2
% while in state 2 because the action will never be chosen.  This is because
% the transition probabilities are identical to those of action 1, but the
% large setup cost is not incurred with action 1.

p(1,3,1) = G(1)*G(2)*G(3);
p(1,3,2) = G(1)*G(2)*(1-G(3));
p(1,3,3) = G(1)*(1-G(2))*G(3);
```

```
p(1,3,4) = G(1)*(1-G(2))*(1-G(3));
p(1,3,5) = (1-G(1))*G(2)*G(3);
p(1,3,6) = (1-G(1))*G(2)*(1-G(3));
p(1,3,7) = (1-G(1))*(1-G(2))*G(3);
p(1,3,8) = (1-G(1))*(1-G(2))*(1-G(3));

p(2,1,2) = R(1)*R(2);
p(2,1,4) = R(1)*F(2);
p(2,1,6) = F(1)*R(2);
p(2,1,8) = F(1)*F(2);

p(2,2,1) = R(1)*R(2)*M(3);
p(2,2,2) = R(1)*R(2)*(1-M(3));
p(2,2,3) = R(1)*F(2)*M(3);
p(2,2,4) = R(1)*F(2)*(1-M(3));
p(2,2,5) = F(1)*R(2)*M(3);
p(2,2,6) = F(1)*R(2)*(1-M(3));
p(2,2,7) = F(1)*F(2)*M(3);
p(2,2,8) = F(1)*F(2)*(1-M(3));

p(2,3,1) = G(1)*G(2)*G(3);
p(2,3,2) = G(1)*G(2)*(1-G(3));
p(2,3,3) = G(1)*(1-G(2))*G(3);
p(2,3,4) = G(1)*(1-G(2))*(1-G(3));
p(2,3,5) = (1-G(1))*G(2)*G(3);
p(2,3,6) = (1-G(1))*G(2)*(1-G(3));
p(2,3,7) = (1-G(1))*(1-G(2))*G(3);
p(2,3,8) = (1-G(1))*(1-G(2))*(1-G(3));

p(3,1,3) = R(1)*R(3);
p(3,1,4) = R(1)*F(3);
p(3,1,7) = F(1)*R(3);
p(3,1,8) = F(1)*F(3);

p(3,2,1) = R(1)*M(2)*R(3);
p(3,2,2) = R(1)*M(2)*F(3);
p(3,2,3) = R(1)*(1-M(2))*R(3);
p(3,2,4) = R(1)*(1-M(2))*F(3);
p(3,2,5) = F(1)*M(2)*R(3);
p(3,2,6) = F(1)*M(2)*F(3);
p(3,2,7) = F(1)*(1-M(2))*R(3);
p(3,2,8) = F(1)*(1-M(2))*F(3);
```

```
p(3,3,1) = G(1)*G(2)*G(3);
p(3,3,2) = G(1)*G(2)*(1-G(3));
p(3,3,3) = G(1)*(1-G(2))*G(3);
p(3,3,4) = G(1)*(1-G(2))*(1-G(3));
p(3,3,5) = (1-G(1))*G(2)*G(3);
p(3,3,6) = (1-G(1))*G(2)*(1-G(3));
p(3,3,7) = (1-G(1))*(1-G(2))*G(3);
p(3,3,8) = (1-G(1))*(1-G(2))*(1-G(3));

p(4,1,4) = R(1);
p(4,1,8) = F(1);

p(4,2,1) = R(1)*M(2)*M(3);
p(4,2,2) = R(1)*M(2)*(1-M(3));
p(4,2,3) = R(1)*(1-M(2))*M(3);
p(4,2,4) = R(1)*(1-M(2))*(1-M(3));
p(4,2,5) = F(1)*M(2)*M(3);
p(4,2,6) = F(1)*M(2)*(1-M(3));
p(4,2,7) = F(1)*(1-M(2))*M(3);
p(4,2,8) = F(1)*(1-M(2))*(1-M(3));

p(4,3,1) = G(1)*G(2)*G(3);
p(4,3,2) = G(1)*G(2)*(1-G(3));
p(4,3,3) = G(1)*(1-G(2))*G(3);
p(4,3,4) = G(1)*(1-G(2))*(1-G(3));
p(4,3,5) = (1-G(1))*G(2)*G(3);
p(4,3,6) = (1-G(1))*G(2)*(1-G(3));
p(4,3,7) = (1-G(1))*(1-G(2))*G(3);
p(4,3,8) = (1-G(1))*(1-G(2))*(1-G(3));

p(5,1,5) = R(2)*R(3);
p(5,1,6) = R(2)*F(3);
p(5,1,7) = F(2)*R(3);
p(5,1,8) = F(2)*F(3);

p(5,2,1) = M(1)*R(2)*R(3);
p(5,2,2) = M(1)*R(2)*F(3);
p(5,2,3) = M(1)*F(2)*R(3);
p(5,2,4) = M(1)*F(2)*F(3);
p(5,2,5) = (1-M(1))*R(2)*R(3);
p(5,2,6) = (1-M(1))*R(2)*F(3);
p(5,2,7) = (1-M(1))*F(2)*R(3);
p(5,2,8) = (1-M(1))*F(2)*F(3);
```

```
p(5,3,1) = G(1)*G(2)*G(3);
p(5,3,2) = G(1)*G(2)*(1-G(3));
p(5,3,3) = G(1)*(1-G(2))*G(3);
p(5,3,4) = G(1)*(1-G(2))*(1-G(3));
p(5,3,5) = (1-G(1))*G(2)*G(3);
p(5,3,6) = (1-G(1))*G(2)*(1-G(3));
p(5,3,7) = (1-G(1))*(1-G(2))*G(3);
p(5,3,8) = (1-G(1))*(1-G(2))*(1-G(3));

p(6,1,6) = R(2);
p(6,1,8) = F(2);

p(6,2,1) = M(1)*R(2)*M(3);
p(6,2,2) = M(1)*R(2)*(1-M(3));
p(6,2,3) = M(1)*F(2)*M(3);
p(6,2,4) = M(1)*F(2)*(1-M(3));
p(6,2,5) = (1-M(1))*R(2)*M(3);
p(6,2,6) = (1-M(1))*R(2)*(1-M(3));
p(6,2,7) = (1-M(1))*F(2)*M(3);
p(6,2,8) = (1-M(1))*F(2)*(1-M(3));

p(6,3,1) = G(1)*G(2)*G(3);
p(6,3,2) = G(1)*G(2)*(1-G(3));
p(6,3,3) = G(1)*(1-G(2))*G(3);
p(6,3,4) = G(1)*(1-G(2))*(1-G(3));
p(6,3,5) = (1-G(1))*G(2)*G(3);
p(6,3,6) = (1-G(1))*G(2)*(1-G(3));
p(6,3,7) = (1-G(1))*(1-G(2))*G(3);
p(6,3,8) = (1-G(1))*(1-G(2))*(1-G(3));

p(7,1,7) = R(3);
p(7,1,8) = F(3);

p(7,2,1) = M(1)*M(2)*R(3);
p(7,2,2) = M(1)*M(2)*F(3);
p(7,2,3) = M(1)*(1-M(2))*R(3);
p(7,2,4) = M(1)*(1-M(2))*F(3);
p(7,2,5) = (1-M(1))*M(2)*R(3);
p(7,2,6) = (1-M(1))*M(2)*F(3);
p(7,2,7) = (1-M(1))*(1-M(2))*R(3);
p(7,2,8) = (1-M(1))*(1-M(2))*F(3);
```

```
p(7,3,1) = G(1)*G(2)*G(3);
p(7,3,2) = G(1)*G(2)*(1-G(3));
p(7,3,3) = G(1)*(1-G(2))*G(3);
p(7,3,4) = G(1)*(1-G(2))*(1-G(3));
p(7,3,5) = (1-G(1))*G(2)*G(3);
p(7,3,6) = (1-G(1))*G(2)*(1-G(3));
p(7,3,7) = (1-G(1))*(1-G(2))*G(3);
p(7,3,8) = (1-G(1))*(1-G(2))*(1-G(3));

p(8,1,8) = 1;

p(8,2,1) = M(1)*M(2)*M(3);
p(8,2,2) = M(1)*M(2)*(1-M(3));
p(8,2,3) = M(1)*(1-M(2))*M(3);
p(8,2,4) = M(1)*(1-M(2))*(1-M(3));
p(8,2,5) = (1-M(1))*M(2)*M(3);
p(8,2,6) = (1-M(1))*M(2)*(1-M(3));
p(8,2,7) = (1-M(1))*(1-M(2))*M(3);
p(8,2,8) = (1-M(1))*(1-M(2))*(1-M(3));

p(8,3,1) = G(1)*G(2)*G(3);
p(8,3,2) = G(1)*G(2)*(1-G(3));
p(8,3,3) = G(1)*(1-G(2))*G(3);
p(8,3,4) = G(1)*(1-G(2))*(1-G(3));
p(8,3,5) = (1-G(1))*G(2)*G(3);
p(8,3,6) = (1-G(1))*G(2)*(1-G(3));
p(8,3,7) = (1-G(1))*(1-G(2))*G(3);
p(8,3,8) = (1-G(1))*(1-G(2))*(1-G(3));

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Define the Immediate Rewards - deterministic part of the reward
% function ra{a}(s) ==> Immediate reward when choosing action a in
% state s

ra = cell(actions);

ra{1} = zeros(TotStates);
ra{2} = zeros(TotStates);
ra{3} = zeros(TotStates);

ra{2}(1) = NA;
ra{2}(2) = C_m(2);
ra{2}(3) = C_m(3);
```

```
ra{2}(4) = C_m(4);
ra{2}(5) = C_m(5);
ra{2}(6) = C_m(6);
ra{2}(7) = C_m(7);
ra{2}(8) = C_m(8);

ra{3}(1) = C_s;
ra{3}(2) = C_s;
ra{3}(3) = C_s;
ra{3}(4) = C_s;
ra{3}(5) = C_s;
ra{3}(6) = C_s;
ra{3}(7) = C_s;
ra{3}(8) = C_s;

% Define the Expected Rewards
% r(s,a) ==> The expected reward when in state s and choosing
% action a. This is equivalent to the immediate reward plus the
% expected penalty cost over the next period.
r = zeros(TotStates,actions);

for s = 1:TotStates
    for a = 1:actions

        % Calculate the expected penalty cost over the next period
        % (the sum of the penalty costs times the probability that
        % that cost is incurred).
        Expected_Penalty = 0;

        for j = 1:TotStates
            Expected_Penalty = Expected_Penalty + p(s,a,j)*C_p(j);
        end

        % Assign the expected reward resulting from taking action a
        % in state s.
        if Expected_Penalty == 0
            r(s,a) = NA; % The action is not feasible
        else
            r(s,a) = ra{a}(s) + Expected_Penalty;
        end
    end
end
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Backward Induction Code (Directly adapted from Capt Sumter's
% code)

% Define the u, ustar and dstar vector
u = zeros(TotStates,actions,N);
ustar = zeros(TotStates,N);
dstar = zeros(TotStates,N);
% Note: The value of any action in the final time period has zero
% reward and therefore zero utility
% ustar(s,N) = 0 for all s

n = N - 1;
while n >= 1
    for s = 1:TotStates
        ustar(s,n) = -realmax;
        for a = 1:actions
            if r(s,a) ~= NA  % Infeasible action - no reward
                expected_value = 0;
                for j = 1:TotStates
                    expected_value = expected_value + p(s,a,j)*ustar(j,n+1);
                end
                u(s,a,n) = r(s,a) + expected_value;

                if u(s,a,n) > ustar(s,n)
                    ustar(s,n) = u(s,a,n);
                    dstar(s,n) = a;
                end
            end

        end
    end

    n = n - 1;   % Decrement the time
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Store the optimal value results (ustar) in the matrix 'results'
results(C_L,C_V) = ustar(1,1);

% Store the optimal policies (dstar) in the cell array "policies"
policies{C_L,C_V} = dstar;
```

```
        clear ustar;
        clear dstar;

    end % end the repair cost loop
end  % end the replacement cost loop

% Open file to write results to
fid = fopen('One_Sat_Ratio.txt','w');

% Write the results
for j=1:C_V
    fprintf(fid, [num2str((101-j)/100),'%']);
end fprintf(fid, '\n'); for i = 1:C_L;
    for j=1:C_V
        fprintf(fid, '%8.5f\t',results(i,j));
    end
    fprintf(fid, '\n');
end

for j=1:C_L
    fprintf(fid, '\n Satellite replacement cost - ');
    fprintf(fid, '%g\t', 50*j);
    fprintf(fid, '\n');
    for i = 1:C_V
        fprintf(fid, 'On-Orbit cost as fraction of replacement cost - ');
        fprintf(fid, [num2str((101-i)/100),'%']);
        fprintf(fid, '\n');
        for s=1:TotStates
            for n=1:N
                fprintf(fid, '%g\t', policies{j,i}(s,n));
            end
            fprintf(fid, '\n');
        end
    end
end
% Close file
fclose(fid);
```

Appendix F. Three-Satellite Two-Way Analysis

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Author: 1Lt Tim Cook
%         AFIT/ENS/GOR-05M
%         March 2005
% This program performs a two-way sensitivity analysis on a Markov decision
% process formulated to find an optimal maintenance policy for a three
% satellite constellation.  In this file, the parameters that are modified
% are the repair costs and the replacement costs.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Define Parameters
N = 20;      % Length of planning horizon
M = [2 2 3];% Number of functions desired operational in each satellite
K = 3;       % Number of Satellites

% Create a vector that keeps the number of degradation levels for each
% satellite
states = zeros(K); for i=1:K
    states(i) = 2^M(i); % Number of possible states for satellite i
end

% Define the total number of states that the entire satellite
% constellation can possibly enter into
TotStates = prod(states(:,1));
% Number of possible actions for each satellite
actions = 3;

% Failure rates -- 1/(expected lifetime in years)*(4 quarters)
lambda1 = 1/(5.5*4);    % Failure rate of function 1
lambda2 = 1/(5.25*4);   % Failure rate of function 2
lambda3 = 1/(6.5*4);    % Failure rate of function 3

% Allocate memory for the matrices that will store the sensitivity
% results before they are output to a file
results = zeros(TotStates, 100); policies = cell(100);

% Use when an action is not feasible for that state
NA = -99999;

% Failure probabilities for each function
F = [1-exp(-lambda1) 1-exp(-lambda2) 1-exp(-lambda3)];
```

```
% Survival probabilities for each function
R = [exp(-lambda1) exp(-lambda2) exp(-lambda3)];
% Probability of a successful repair for each function
M = [.95 .96 .97];
% Probability of a successful replacement for each function
G = [.975 .94 .98];

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Define Transition Probabilities
% p{k}(s,a,j) ==> Probability  for satellite k of going to state j,
%                 given presently in state s and choosing action a for
%                 that individual satellite
% Allocate memory for the transition probability matricies
% using cell arrays
p = cell(K); for i = 1:K
    p(i) = {zeros(states(i), actions, states(i))};
end

% Define the transition probabilities
p{1}(1,1,1) = R(1)*R(2);
p{1}(1,1,2) = R(1)*F(2);
p{1}(1,1,3) = F(1)*R(2);
p{1}(1,1,4) = F(1)*F(2);

% There is no need to include the transition probabilities for action 2
% while in state 2 because the action will never be chosen.  This is because
% the transition probabilities are identical to those of action 1, but the
% large setup cost is not incurred with action 1.  The same holds for the other
% two satellites.

p{1}(1,3,1) = G(1)*G(2);
p{1}(1,3,2) = G(1)*(1-G(2));
p{1}(1,3,3) = (1-G(1))*G(2);
p{1}(1,3,4) = (1-G(1))*(1-G(2));

p{1}(2,1,2) = R(1);
p{1}(2,1,4) = F(1);

p{1}(2,2,1) = R(1)*M(2);
p{1}(2,2,2) = R(1)*(1-M(2));
p{1}(2,2,3) = F(1)*M(2);
p{1}(2,2,4) = F(1)*(1-M(2));
```

```
p{1}(2,3,1) = G(1)*G(2);
p{1}(2,3,2) = G(1)*(1-G(2));
p{1}(2,3,3) = (1-G(1))*G(2);
p{1}(2,3,4) = (1-G(1))*(1-G(2));

p{1}(3,1,3) = R(2);
p{1}(3,1,4) = F(2);

p{1}(3,2,1) = M(1)*R(2);
p{1}(3,2,2) = M(1)*F(2);
p{1}(3,2,3) = (1-M(1))*R(2);
p{1}(3,2,4) = (1-M(1))*F(2);

p{1}(3,3,1) = G(1)*G(2);
p{1}(3,3,2) = G(1)*(1-G(2));
p{1}(3,3,3) = (1-G(1))*G(2);
p{1}(3,3,4) = (1-G(1))*(1-G(2));

p{1}(4,1,4) = 1;

p{1}(4,2,1) = M(1)*M(2);
p{1}(4,2,2) = M(1)*(1-M(2));
p{1}(4,2,3) = (1-M(1))*M(2);
p{1}(4,2,4) = (1-M(1))*(1-M(2));

p{1}(4,3,1) = G(1)*G(2);
p{1}(4,3,2) = G(1)*(1-G(2));
p{1}(4,3,3) = (1-G(1))*G(2);
p{1}(4,3,4) = (1-G(1))*(1-G(2));

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

p{2}(1,1,1) = R(1)*R(2);
p{2}(1,1,2) = R(1)*F(2);
p{2}(1,1,3) = F(1)*R(2);
p{2}(1,1,4) = F(1)*F(2);

p{2}(1,3,1) = G(1)*G(2);
p{2}(1,3,2) = G(1)*(1-G(2));
p{2}(1,3,3) = (1-G(1))*G(2);
p{2}(1,3,4) = (1-G(1))*(1-G(2));

p{2}(2,1,2) = R(1);
```

```
p{2}(2,1,4) = F(1);

p{2}(2,2,1) = R(1)*M(2);
p{2}(2,2,2) = R(1)*(1-M(2));
p{2}(2,2,3) = F(1)*M(2);
p{2}(2,2,4) = F(1)*(1-M(2));

p{2}(2,3,1) = G(1)*G(2);
p{2}(2,3,2) = G(1)*(1-G(2));
p{2}(2,3,3) = (1-G(1))*G(2);
p{2}(2,3,4) = (1-G(1))*(1-G(2));

p{2}(3,1,3) = R(2);
p{2}(3,1,4) = F(2);

p{2}(3,2,1) = M(1)*R(2);
p{2}(3,2,2) = M(1)*F(2);
p{2}(3,2,3) = (1-M(1))*R(2);
p{2}(3,2,4) = (1-M(1))*F(2);

p{2}(3,3,1) = G(1)*G(2);
p{2}(3,3,2) = G(1)*(1-G(2));
p{2}(3,3,3) = (1-G(1))*G(2);
p{2}(3,3,4) = (1-G(1))*(1-G(2));

p{2}(4,1,4) = 1;

p{2}(4,2,1) = M(1)*M(2);
p{2}(4,2,2) = M(1)*(1-M(2));
p{2}(4,2,3) = (1-M(1))*M(2);
p{2}(4,2,4) = (1-M(1))*(1-M(2));

p{2}(4,3,1) = G(1)*G(2);
p{2}(4,3,2) = G(1)*(1-G(2));
p{2}(4,3,3) = (1-G(1))*G(2);
p{2}(4,3,4) = (1-G(1))*(1-G(2));

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

p{3}(1,1,1) = R(1)*R(2)*R(3); p{3}(1,1,2) = R(1)*R(2)*F(3);
p{3}(1,1,3) = R(1)*F(2)*R(3); p{3}(1,1,4) = R(1)*F(2)*F(3);
p{3}(1,1,5) = F(1)*R(2)*R(3); p{3}(1,1,6) = F(1)*R(2)*F(3);
p{3}(1,1,7) = F(1)*F(2)*R(3); p{3}(1,1,8) = F(1)*F(2)*F(3);
```

```
p{3}(1,3,1) = G(1)*G(2)*G(3);
p{3}(1,3,2) = G(1)*G(2)*(1-G(3));
p{3}(1,3,3) = G(1)*(1-G(2))*G(3);
p{3}(1,3,4) = G(1)*(1-G(2))*(1-G(3));
p{3}(1,3,5) = (1-G(1))*G(2)*G(3);
p{3}(1,3,6) = (1-G(1))*G(2)*(1-G(3));
p{3}(1,3,7) = (1-G(1))*(1-G(2))*G(3);
p{3}(1,3,8) = (1-G(1))*(1-G(2))*(1-G(3));

p{3}(2,1,2) = R(1)*R(2);
p{3}(2,1,4) = R(1)*F(2);
p{3}(2,1,6) = F(1)*R(2);
p{3}(2,1,8) = F(1)*F(2);

p{3}(2,2,1) = R(1)*R(2)*M(3);
p{3}(2,2,2) = R(1)*R(2)*(1-M(3));
p{3}(2,2,3) = R(1)*F(2)*M(3);
p{3}(2,2,4) = R(1)*F(2)*(1-M(3));
p{3}(2,2,5) = F(1)*R(2)*M(3);
p{3}(2,2,6) = F(1)*R(2)*(1-M(3));
p{3}(2,2,7) = F(1)*F(2)*M(3);
p{3}(2,2,8) = F(1)*F(2)*(1-M(3));

p{3}(2,3,1) = G(1)*G(2)*G(3);
p{3}(2,3,2) = G(1)*G(2)*(1-G(3));
p{3}(2,3,3) = G(1)*(1-G(2))*G(3);
p{3}(2,3,4) = G(1)*(1-G(2))*(1-G(3));
p{3}(2,3,5) = (1-G(1))*G(2)*G(3);
p{3}(2,3,6) = (1-G(1))*G(2)*(1-G(3));
p{3}(2,3,7) = (1-G(1))*(1-G(2))*G(3);
p{3}(2,3,8) = (1-G(1))*(1-G(2))*(1-G(3));

p{3}(3,1,3) = R(1)*R(3);
p{3}(3,1,4) = R(1)*F(3);
p{3}(3,1,7) = F(1)*R(3);
p{3}(3,1,8) = F(1)*F(3);

p{3}(3,2,1) = R(1)*M(2)*R(3);
p{3}(3,2,2) = R(1)*M(2)*F(3);
p{3}(3,2,3) = R(1)*(1-M(2))*R(3);
p{3}(3,2,4) = R(1)*(1-M(2))*F(3);
p{3}(3,2,5) = F(1)*M(2)*R(3);
```

```
p{3}(3,2,6) = F(1)*M(2)*F(3);
p{3}(3,2,7) = F(1)*(1-M(2))*R(3);
p{3}(3,2,8) = F(1)*(1-M(2))*F(3);

p{3}(3,3,1) = G(1)*G(2)*G(3);
p{3}(3,3,2) = G(1)*G(2)*(1-G(3));
p{3}(3,3,3) = G(1)*(1-G(2))*G(3);
p{3}(3,3,4) = G(1)*(1-G(2))*(1-G(3));
p{3}(3,3,5) = (1-G(1))*G(2)*G(3);
p{3}(3,3,6) = (1-G(1))*G(2)*(1-G(3));
p{3}(3,3,7) = (1-G(1))*(1-G(2))*G(3);
p{3}(3,3,8) = (1-G(1))*(1-G(2))*(1-G(3));

p{3}(4,1,4) = R(1);
p{3}(4,1,8) = F(1);

p{3}(4,2,1) = R(1)*M(2)*M(3);
p{3}(4,2,2) = R(1)*M(2)*(1-M(3));
p{3}(4,2,3) = R(1)*(1-M(2))*M(3);
p{3}(4,2,4) = R(1)*(1-M(2))*(1-M(3));
p{3}(4,2,5) = F(1)*M(2)*M(3);
p{3}(4,2,6) = F(1)*M(2)*(1-M(3));
p{3}(4,2,7) = F(1)*(1-M(2))*M(3);
p{3}(4,2,8) = F(1)*(1-M(2))*(1-M(3));

p{3}(4,3,1) = G(1)*G(2)*G(3);
p{3}(4,3,2) = G(1)*G(2)*(1-G(3));
p{3}(4,3,3) = G(1)*(1-G(2))*G(3);
p{3}(4,3,4) = G(1)*(1-G(2))*(1-G(3));
p{3}(4,3,5) = (1-G(1))*G(2)*G(3);
p{3}(4,3,6) = (1-G(1))*G(2)*(1-G(3));
p{3}(4,3,7) = (1-G(1))*(1-G(2))*G(3);
p{3}(4,3,8) = (1-G(1))*(1-G(2))*(1-G(3));

p{3}(5,1,5) = R(2)*R(3);
p{3}(5,1,6) = R(2)*F(3);
p{3}(5,1,7) = F(2)*R(3);
p{3}(5,1,8) = F(2)*F(3);

p{3}(5,2,1) = M(1)*R(2)*R(3);
p{3}(5,2,2) = M(1)*R(2)*F(3);
p{3}(5,2,3) = M(1)*F(2)*R(3);
p{3}(5,2,4) = M(1)*F(2)*F(3);
```

```
p{3}(5,2,5) = (1-M(1))*R(2)*R(3);
p{3}(5,2,6) = (1-M(1))*R(2)*F(3);
p{3}(5,2,7) = (1-M(1))*F(2)*R(3);
p{3}(5,2,8) = (1-M(1))*F(2)*F(3);

p{3}(5,3,1) = G(1)*G(2)*G(3);
p{3}(5,3,2) = G(1)*G(2)*(1-G(3));
p{3}(5,3,3) = G(1)*(1-G(2))*G(3);
p{3}(5,3,4) = G(1)*(1-G(2))*(1-G(3));
p{3}(5,3,5) = (1-G(1))*G(2)*G(3);
p{3}(5,3,6) = (1-G(1))*G(2)*(1-G(3));
p{3}(5,3,7) = (1-G(1))*(1-G(2))*G(3);
p{3}(5,3,8) = (1-G(1))*(1-G(2))*(1-G(3));

p{3}(6,1,6) = R(2);
p{3}(6,1,8) = F(2);

p{3}(6,2,1) = M(1)*R(2)*M(3);
p{3}(6,2,2) = M(1)*R(2)*(1-M(3));
p{3}(6,2,3) = M(1)*F(2)*M(3);
p{3}(6,2,4) = M(1)*F(2)*(1-M(3));
p{3}(6,2,5) = (1-M(1))*R(2)*M(3);
p{3}(6,2,6) = (1-M(1))*R(2)*(1-M(3));
p{3}(6,2,7) = (1-M(1))*F(2)*M(3);
p{3}(6,2,8) = (1-M(1))*F(2)*(1-M(3));

p{3}(6,3,1) = G(1)*G(2)*G(3);
p{3}(6,3,2) = G(1)*G(2)*(1-G(3));
p{3}(6,3,3) = G(1)*(1-G(2))*G(3);
p{3}(6,3,4) = G(1)*(1-G(2))*(1-G(3));
p{3}(6,3,5) = (1-G(1))*G(2)*G(3);
p{3}(6,3,6) = (1-G(1))*G(2)*(1-G(3));
p{3}(6,3,7) = (1-G(1))*(1-G(2))*G(3);
p{3}(6,3,8) = (1-G(1))*(1-G(2))*(1-G(3));

p{3}(7,1,7) = R(3);
p{3}(7,1,8) = F(3);

p{3}(7,2,1) = M(1)*M(2)*R(3);
p{3}(7,2,2) = M(1)*M(2)*F(3);
p{3}(7,2,3) = M(1)*(1-M(2))*R(3);
p{3}(7,2,4) = M(1)*(1-M(2))*F(3);
p{3}(7,2,5) = (1-M(1))*M(2)*R(3);
```

```
p{3}(7,2,6) = (1-M(1))*M(2)*F(3);
p{3}(7,2,7) = (1-M(1))*(1-M(2))*R(3);
p{3}(7,2,8) = (1-M(1))*(1-M(2))*F(3);

p{3}(7,3,1) = G(1)*G(2)*G(3);
p{3}(7,3,2) = G(1)*G(2)*(1-G(3));
p{3}(7,3,3) = G(1)*(1-G(2))*G(3);
p{3}(7,3,4) = G(1)*(1-G(2))*(1-G(3));
p{3}(7,3,5) = (1-G(1))*G(2)*G(3);
p{3}(7,3,6) = (1-G(1))*G(2)*(1-G(3));
p{3}(7,3,7) = (1-G(1))*(1-G(2))*G(3);
p{3}(7,3,8) = (1-G(1))*(1-G(2))*(1-G(3));

p{3}(8,1,8) = 1;

p{3}(8,2,1) = M(1)*M(2)*M(3);
p{3}(8,2,2) = M(1)*M(2)*(1-M(3));
p{3}(8,2,3) = M(1)*(1-M(2))*M(3);
p{3}(8,2,4) = M(1)*(1-M(2))*(1-M(3));
p{3}(8,2,5) = (1-M(1))*M(2)*M(3);
p{3}(8,2,6) = (1-M(1))*M(2)*(1-M(3));
p{3}(8,2,7) = (1-M(1))*(1-M(2))*M(3);
p{3}(8,2,8) = (1-M(1))*(1-M(2))*(1-M(3));

p{3}(8,3,1) = G(1)*G(2)*G(3);
p{3}(8,3,2) = G(1)*G(2)*(1-G(3));
p{3}(8,3,3) = G(1)*(1-G(2))*G(3);
p{3}(8,3,4) = G(1)*(1-G(2))*(1-G(3));
p{3}(8,3,5) = (1-G(1))*G(2)*G(3);
p{3}(8,3,6) = (1-G(1))*G(2)*(1-G(3));
p{3}(8,3,7) = (1-G(1))*(1-G(2))*G(3);
p{3}(8,3,8) = (1-G(1))*(1-G(2))*(1-G(3));

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Define the state space for each satellite
Sat_s = cell(K); Sat_s{1} = [1 1
    1 0
    0 1
    0 0];
Sat_s{2} = [1 1
    1 0
    0 1
    0 0];
```

```
Sat_s{3} = [1 1 1
    1 1 0
    1 0 1
    1 0 0
    0 1 1
    0 1 0
    0 0 1
    0 0 0];

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Define the overall state space for the constellation
Constellation_s = cell(TotStates); s1 = 1; s2 = 1; s3 = 1;

for i = 1:TotStates
    Constellation_s{i} = [s1 s2 s3];
    s3 = s3 + 1;
    if s3 > states(3)
        s3 = 1;
        s2 = s2 + 1;
    end
    if s2 > states(2)
        s2 = 1;
        s1 = s1 + 1;
    end
    if s1 > states(1)
        s1 = 1;
    end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Penalty Cost assigned per state
% The decision maker desires that
C_p = zeros(TotStates); for i = 1:TotStates

    % Function 1 Penalty Cost
    s1 = Constellation_s{i}(1);
    s2 = Constellation_s{i}(2);
    s3 = Constellation_s{i}(3);
    sum = Sat_s{1}(s1,1)+Sat_s{2}(s2,1)+Sat_s{3}(s3,1);
    if sum == 3
        C_p(i) = C_p(i) + 0;
    elseif sum == 2
        C_p(i) = C_p(i) + 0;
    elseif sum == 1
```

```
        C_p(i) = C_p(i) - 500;
    elseif sum == 0
        C_p(i) = C_p(i) - 650;
    end

    % Function 2 Penalty Cost
    sum = Sat_s{1}(s1,2)+Sat_s{2}(s2,2)+Sat_s{3}(s3,2);
    if sum == 3
        C_p(i) = C_p(i) + 0;
    elseif sum == 2
        C_p(i) = C_p(i) + 0;
    elseif sum == 1
        C_p(i) = C_p(i) - 450;
    elseif sum == 0
        C_p(i) = C_p(i) - 700;
    end

    % Function 3 Penalty Cost
    if Sat_s{3}(s3,3) == 1
        C_p(i) = C_p(i) + 0;
    elseif Sat_s{3}(s3,3) == 0
        C_p(i) = C_p(i) - 200;
    end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Define the overall constellation action set in a cell array
Constellation_a = cell(actions^K); a1=1; a2=1; a3=1;

for i = 1:actions^K
    Constellation_a{i}=[a1 a2 a3];
    a3 = a3 + 1;
    if a3 > actions
        a2 = a2 + 1;
        a3 = 1;
    end
    if a2 > actions
        a1 = a1 + 1;
        a2 = 1;
    end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
% Define the overall constellation transition probabilities.  Do this
% by looping through each possibile (s,a,j) combination in the
% constellation and multiplying the individual satellite transition
% probabilities that would lead to the overall satellite constellation
% transition.  The user of this code must know clearly what state the
% index of each satellite's state and action vector represents.
s1 = 1;
s2 = 1;
s3 = 1;
a1 = 1;
a2 = 1;
a3 = 1;
j1 = 1;
j2 = 1;
j3 = 1;
Constellation_p = zeros(TotStates,actions^K,TotStates);

for s = 1:TotStates
    for a = 1:actions^K
        for j = 1:TotStates
            Constellation_p(s,a,j) = p{1}(s1,a1,j1)*p{2}(s2,a2,j2)*p{3}(s3,a3,j3);

            % Eliminate transition probabilities associated with
            % infeasible actions. Set the transition probabilities to
            % 0 for all infeasible actions and therefore sets the
            % expected rewards to 0 so they can be identified and then
            % set to NA.
            if s1 == 1
                if a1 == 2
                    Constellation_p(s,a,j) = 0;
                end
            end
            if s2 == 1
                if a2 == 2
                    Constellation_p(s,a,j) = 0;
                end
            end
            if s3 == 1
                if a3 == 2
                    Constellation_p(s,a,j) = 0;
                end
            end
            % The previous block eliminates any on orbit repairs to a
```

```
% satellite that has nothing broken.

if s1 == states(1)
    if s2 == states(2)
        if s3 == states(3)
            if a1 == 1
                if a2 == 1
                    if a3 == 1
                        Constellation_p(s,a,j) = 0;
                    end
                end
            end
        end
    end
end
% The previous block eliminates the possibility of doing
% nothing when all functions of all satellites are
% non-operational.

j3 = j3 + 1;
if j3 > states(3)
    j3 = 1;
    j2 = j2 + 1;
end
if j2 > states(2)
    j2 = 1;
    j1 = j1 + 1;
end
if j1 > states(1)
    j1 = 1;
    a3 = a3 + 1;
end
if a3 > actions
    a3 = 1;
    a2 = a2 + 1;
end
if a2 > actions
    a2 = 1;
    a1 = a1 + 1;
end
if a1 > actions
    a1 = 1;
    s3 = s3 + 1;
```

```
            end
            if s3 > states(3)
                s3 = 1;
                s2 = s2 + 1;
            end
            if s2 > states(2)
                s2 = 1;
                s1 = s1 + 1;
            end
            if s1 > states(1)
                s1 = 1;
            end
        end
    end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Sensitivity analysis portion of the code
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Begin the outer for loop that varies the satellite unit cost.
for C_L = 1:20

    % Satellite unit cost
    C_unit = -50*C_L;

    % Begin the inner for loop that varies the space vehicle cost from 0%
    % to 100% of the satellite unit cost.
    for C_V = 1:101
        disp([C_L,C_V])

        % On-orbit repair costs
        SpaceVehicle = -((101-C_V)/100)*C_unit; % a positive number
        disp([C_unit,-SpaceVehicle])

        C_m = cell(K);
        C_m{1} = [-realmax -(SpaceVehicle + 15) -(SpaceVehicle + 35) -(SpaceVehicle + 50)];
        C_m{2} = [-realmax -(SpaceVehicle + 15) -(SpaceVehicle + 35) -(SpaceVehicle + 50)];
        C_m{3} = [-realmax -(SpaceVehicle + 20) -(SpaceVehicle + 15) -(SpaceVehicle + 35) -(SpaceVehicle + 35)
                  -(SpaceVehicle + 55) -(SpaceVehicle + 50) -(SpaceVehicle + 70)];

        % Satellite replacement cost
        C_s = [C_unit-30 C_unit-30 C_unit-50];
```

```
% Define the Immediate Rewards for each individual satellite
% ra(a,s) ==> Immediate reward when choosing action a while in
% state s (constellation).
% Assume that the repair costs are the same for either function on
% all satellites
ra = zeros(actions^K, TotStates);

for Cons_A = 1:actions^K
    for Cons_S = 1:TotStates
        for k = 1:K
            if Constellation_a{Cons_A}(k) == 1
                ra(Cons_A,Cons_S) = ra(Cons_A,Cons_S) + 0;
            end
            if Constellation_a{Cons_A}(k) == 2
                SatState = Constellation_s{Cons_S}(k);
                ra(Cons_A,Cons_S) = ra(Cons_A,Cons_S) + C_m{k}(SatState);
            end
            if Constellation_a{Cons_A}(k) == 3
                ra(Cons_A,Cons_S) = ra(Cons_A,Cons_S) + C_s(k);
            end
        end
    end
end

% Define the Expected Rewards
% r(s,a) ==> The expected reward when in state s and choosing
% action a for the constellation. This is equivalent to the
% immediate reward plus the expected penalty cost over the next
% period.
r = zeros(TotStates,actions^K);

for s = 1:TotStates
    for a = 1:actions^K

        % Calculate the expected penalty cost over the next period
        % (the sum of the penalty costs times the probability that
        % that cost is incurred).
        Expected_penalty = 0;

        for j = 1:TotStates
            Expected_penalty = Expected_penalty + Constellation_p(s,a,j)*C_p(j);
        end
        % Assign the expected reward resulting from taking action a
```

```
        % in state s.
        if Expected_penalty == 0
            r(s,a) = NA;
        else
            r(s,a) = ra(a,s) + Expected_penalty;
        end
    end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Backward Induction Code (Directly adapted from Capt Sumter's code)

% Define the u, ustar and dstar vector
u = zeros(TotStates,actions,N);
ustar = zeros(TotStates,N);
dstar = zeros(TotStates,N);
% Note: The value of any action in the final time period has zero
% reward and therefore zero utility
% ustar(s,N) = 0 for all s

n = N - 1;
while n >= 1
    for s = 1:TotStates  % Loop through all constellation states
        ustar(s,n) = -realmax;
        for a = 1:actions^K  % Loop through all actions
            if r(s,a) ~= NA  % Infeasible action - no reward

                expected_value = 0;
                for j = 1:TotStates
                    expected_value = expected_value + Constellation_p(s,a,j)*ustar(j,n+1);
                end % End For Loop for expected value
                u(s,a,n) = r(s,a) + expected_value;

                if u(s,a,n) > ustar(s,n)
                    ustar(s,n) = u(s,a,n);
                    dstar(s,n) = a;
                end
            end
        end
    end
    n = n - 1;
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
        % Store the optimal value results (ustar) in the matrix 'results'
        results(C_L,C_V) = ustar(1,1);

        % Store the optimal policies (dstar) in the cell array "policies"
        policies{C_L,C_V} = dstar;

        clear ustar;
        clear dstar;

    end % end the repair cost loop
end  % end the replacement cost loop

% Open file to write results to
fid = fopen('Three_Sat_Ratio.txt','w');

% Write the results
for j=1:C_V
    fprintf(fid, [num2str((101-j)/100),'%']);
end fprintf(fid, '\n'); for i = 1:C_L;
    for j=1:C_V
        fprintf(fid, '%8.5f\t',results(i,j));
    end
    fprintf(fid, '\n');
end

for j=1:C_L
    fprintf(fid, '\n Satellite replacement cost - ');
    fprintf(fid, '%g\t', 50*j);
    fprintf(fid, '\n');
    for i = 1:C_V
        fprintf(fid, 'On-Orbit cost as fraction of replacement cost - ');
        fprintf(fid, [num2str((101-i)/100),'%']);
        fprintf(fid, '\n');
        for s=1:TotStates
            for n=1:N
                fprintf(fid, '%g\t', policies{j,i}(s,n));
            end
            fprintf(fid, '\n');
        end
    end
end
% Close file
fclose(fid);
```

Bibliography

1. Advanced Industrial Science and Technology. "AIST: Development of a Space Maintenance Robot". http://www.aist.go.jp/aist_e/latest_research/2002/20020523/20020523.html

2. Apgar, H., D. Bearden, and R. Wong (1999). Cost Modeling. *Space Mission Analysis and Design*. 3rd ed., Microcosm Press, El Segundo, California and Kluwer Academic Publishers, Boston, Massachusetts, 783-820.

3. Baker, D. (editor) (2002-2003). *Jane's Space Directory: 2002-2003*. Jane's Information Group Inc. 1340 Braddock Place, Suite 300, Alexandria, VA. 22314-1651.

4. Barlow, R. E. and F. Proschan. (1965) *Mathematical Theory of Reliability*. John Wiley & Sons, Inc., New York.

5. Barlow, R. E. and L. C. Hunter. (1960) Optimum preventive maintenance policies. *Operations Research*, **8**, 90-100.

6. Barlow, R. E., L. C. Hunter, and F. Proschan. (1963) Optimum checking procedures. *Journal of the Society for Industrial and Applied Mathematics*, **4**, 1078-1095.

7. Bloch-Mercier, S. (2002) A preventive maintenance policy with sequential checking procedure for a Markov deteriorating system. *European Journal of Operations Research*, **147**, 548-576.

8. Brown, G., R. Dell, H. Holtz and A. Newman. (2003) How Space Command optimizes long term investment in space systems. *Interfaces*, **33**, (4), 1-14.

9. Campbell, N. R. (1941). The replacement of perishable members of a continually operating system. *J. Roy. Statist. Soc.*, **7**, 110-130.

10. Cho, I.D., and Parlar, M. (1991) A survey of maintenance models for multi-unit systems. *European Journal of Operations Research*, **51**, 1-23.

11. Comstock, J. (2004). Deputy Program Director, Welkin Associates, Ltd., Chantilly, VA. Telephone interview. 28 June 2004.

12. Ebeling, C.E. (1997) *An Introduction to Reliability and Maintainability Engineering*. McGraw-Hill Companies, Inc., Boston, MA.

13. Gertsbakh, I. B. (1984) Optimal group preventive maintenance of a system with observable state parameter. *Advances in Applied Probability*, **16**, 923-925.

14. Grall, A., C. Bérenguer, and L. Dieulle. (2002) A condition-based maintenance policy for stochastically deteriorating systems. *Reliability Engineering and System Safety*, **76**, 167-180.

15. Gürler, Ü. and A. Kaya. (2002) A maintenance policy for a system with multi-state components: an approximate solution. *Reliability Engineering and System Safety*, **76**, 117-127.

16. Hansen, C. K. (1995). Reliability prediction andsimulation for a communications-satellite fleet. *1995 Proceedings, Annual Reliability and Maintainibility Symposium*, Washington D.C., Jan. 16-19, 152-158.

17. Jacobs, J. L., A. J. Amato, M. D. Bolduc, S. P. Butler, and J. E. Caspero (1992). OSCARS - Operational Constellation Availability and Reliability Simulation. *Proceedings of the 1992 Summer Computer Simulation Conference*, Reno,Nevada, July 27-30, 878-882.

18. Kececioglu, D. and F. Sun (1995) A general discrete-time dynamic programming model for the opportunistic replacement policy and its application to ball-bearing systems. *Reliability Engineering and System Safety*, **47**, 175-185.

19. Kulkarni, V. G. (1995) *Modeling and Analysis of Stochastic Systems*. Chapman & Hall/CRC, New York.

20. Launius, R. D. "Sputnik and the Origins of the Space Age" http://www.hq.nasa.gov/office/pao/History/sputnik/sputorig.html

21. Lotka, A. J. (1939) A contribution to the theory of self-renewing aggregates with special reference to industrial replacement. *Annals of Mathematical Statistics*, **10**, 1-25.

22. Morse, P. M. (1958) *Queues, Inventories and Maintenance*. John Wiley & Sons, Inc., New York

23. Moustafa, M.S., E.Y. Abder Maksoud, S. Sadek (2004) Optimal major and minimal maintenance policies for deteriorating systems. *Reliability Engineering and System Safety*, **83**, 363-368.

24. McCall, J. J. (1965) Maintenance policies for stochastically failing equipment: a survey. *Management Science*, **11**, 493-524.

25. NASA Goddard Space Flight Center. "The Hubble Project" http://hubble.nasa.gov/

26. Nummelin, E. (1980) A general failure model: optimal replacement with state dependent replacement and failure costs. *Annals of Mathematical Statistics*, **10**, 1-25.

27. On-Orbit Servicing (OOS). "The one step link to OOS". http://www.on-orbit-servicing.com/

28. Park, K. S. (1988). Optimal wear-limit replacement with wear-dependent failures. *IEEE Transactions on Reliability*, **37**, 293-294.

29. Pham, H., and H. Wang (2000). Optimal (τ, T) opportunistic maintenance of a k-out-of-n:G system with imperfect PM and partial failure. *Naval Research Logistics*, **47**, 223-239.

30. Pierskalla, W.P. and J.A. Voelker (1976) A survey of maintenance models: The control and surveillance of deteriorating systems. *Naval Research Logistics Quarterly*, **23**, 353-388.

31. Preinreich, G. A. D. (1940) Economic life of industrial equipment. *Econometrica*, **8**, 12-44.

32. Puterman, M. L. (1994). *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc., New York.

33. Radner, R, and D. W. Jorgenson (1962). Optimal replacement and inspection of stochastically failing equipment, Chapter 12. *Studies in Applied Probability and Management Science* Edited by Arrow, Karlin, and Scarf. Stanford University Press, Stanford, California.

34. Radner, R., and D. W. Jorgenson (1963). Opportunistic replacement of a single part in the presence of several monitored parts. *Management Science*, **10**, 70-84.

35. Reeves, E. I. (1999). Spacecraft design and sizing. *Space Mission Analysis and Design*. 3rd ed., Microcosm Press, El Segundo, California and Kluwer Academic Publishers, Boston, Massachusetts, 301-352.

36. Sherif, Y. S. and M. L. Smith (1981) Optimal maintenance models for systems subject to failure: A review *Naval Research Logistics Quarterly*, **28**, 47-74.

37. Sparte, Inc. "SEAS - System Effectiveness Analysis Simulation". http://www.teamseas.com/ 14 Jan 2005.

38. Su, C-T, S-C Wu, and C-C Chang (2000). Multiaction maintenance subject to action-dependent risk and stochastic failure. *European Journal of Operations Research*, **125**, 133-148.

39. Sumter, B. R. *Optimal Replacement Policies for Satellite Constellations*. MS Thesis, AFIT/GOR/ENS/02-23. Graduate School of Engineering and Management, Air Force Institute of Technology (AU), Wright-Patterson AFB, OH, March 2003

40. TheStreet.Com. "AT&T Wireless Posts Surprise Profit". http://www.thestreet.com/_tscs/tech/telecom/10172498.html 21 July 2004

41. United States Air Force. "Air Force Link" http://www.af.mil/factsheets/

42. Valdez-Flores, C. and R. M. Feldman (1989). A survey of preventive maintenance models for stochastically deteriorating single-unit systems. *Naval Research Logistics*, **36**, 419-446.

43. Vergin, R. C., and M. Scriabin (1977). Maintenance scheduling for multicomponent equipment. *AIIE*, **9**, 297-305.

44. Wang, H. (2002) A survey of maintenance policies of deteriorating systems. *European Journal of Operations Research*, **139**, 469-489.

45. Wilson, S. T. (1990) *Measuring the Effectiveness of Space: Satellite Weather Systems.* M.S. Thesis, Graduate School of Engineering and Management, Air Force Institute of Technology, Wright-Patterson AFB, Ohio.

REPORT DOCUMENTATION PAGE	*Form Approved* *OMB No. 074-0188*

1. REPORT DATE *(DD-MM-YYYY)*	**2. REPORT TYPE**	**3. DATES COVERED** *(From – To)*
01-03-2005	**Master's Thesis**	Mar 2004 – Mar 2005

4. TITLE AND SUBTITLE

Optimal Maintenance for Stochastically Degrading Satellite Constellations

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Cook, Timothy, J., First Lieutenant, USAF

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAMES(S) AND ADDRESS(S)

Air Force Institute of Technology
Graduate School of Engineering and Management (AFIT/EN)
2950 Hobson Street, Building 642
WPAFB OH 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT/GOR/ENS/05-05

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

National Reconnaissance Office
Attn: William J. Comstock, Rm 43D19H
14675 Leed Road
Chantilly, VA 20151-1715
(703) 808-4436

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

This thesis develops a methodology to determine an optimal policy for maintaining a satellite constellation that degrades over time. Previous work has developed a methodology to compute an optimal replacement policy for a satellite constellation in which satellites were viewed as binary entities, either operational or failed. This research extends the previous models by developing an optimal maintenance policy for satellite constellations in which each satellite may operate in a finite number of degraded states. The constellation is assumed to consist of a finite number of satellites, each with a finite number of functions with distinct failure mechanisms. Assuming each function lifetime is exponentially distributed, the stochastic degradation process is modelled as a discrete-time Markov chain. The degradation process is subsequently used to formulate an optimization problem as a finite planning horizon Markov decision process in which the total expected loss of utility is minimized. The maintenance actions considered include on-orbit repairs and satellite replacements, each of which has an associated level of risk. Numerical examples are presented to illustrate the model, and a parametric sensitivity analysis is performed using notional data to determine conditions for which the resulting policy consists of only replacements, only on-orbit repairs, or mixtures of replacements and on-orbit repairs.

15. SUBJECT TERMS

Satellite Constellations, Optimal Maintenance Policy, Markov Chains, Dynamic Programming, Stochastic Degradation

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON**
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**			Jeffrey P. Kharoufeh, Phd, (ENS)
					19b. TELEPHONE NUMBER *(Include area code)*
U	**U**	**U**	**UU**	189	(937) 785-3636 ext 4603; e-mail: Jeffrey.Kharoufeh@afit.edu